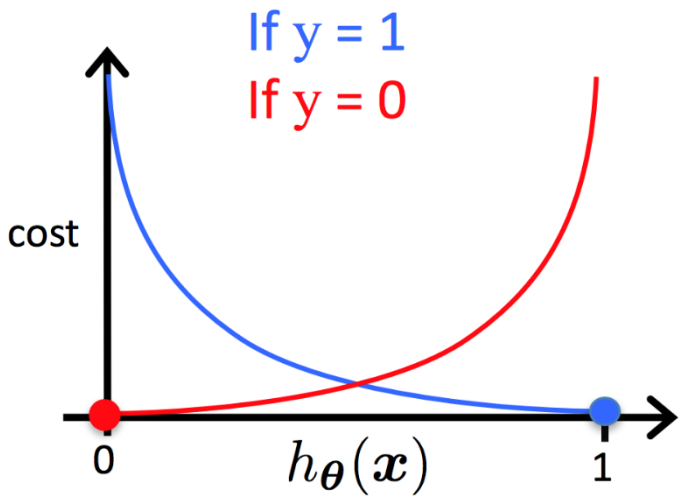
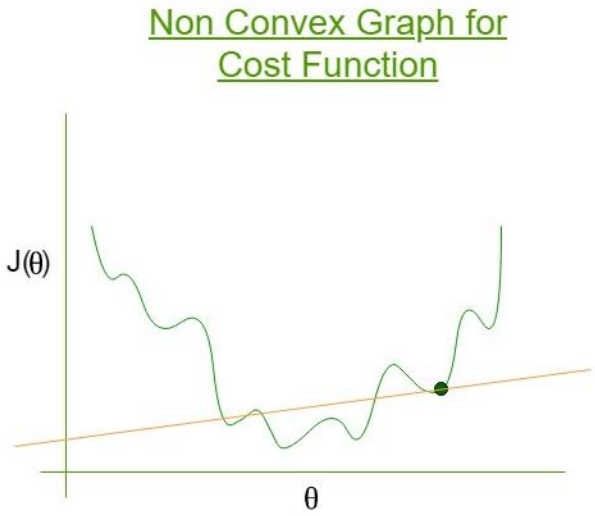
سوال 1-

خیر – با توجه به cost function معمول برای لاجیستیک رگرشن در نهایت نمودار cost function یک نمودار convex خواهد بود پس گرادیانت دیسنت در global minimum متوقف خواهد شد.

Cost function = $$-\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log\left(f_{\vec{w},b}(\vec{x}^{(i)})\right)+(1-y^{(i)})\log\left(1-f_{\vec{w},b}(\vec{x}^{(i)})\right)\right]$$

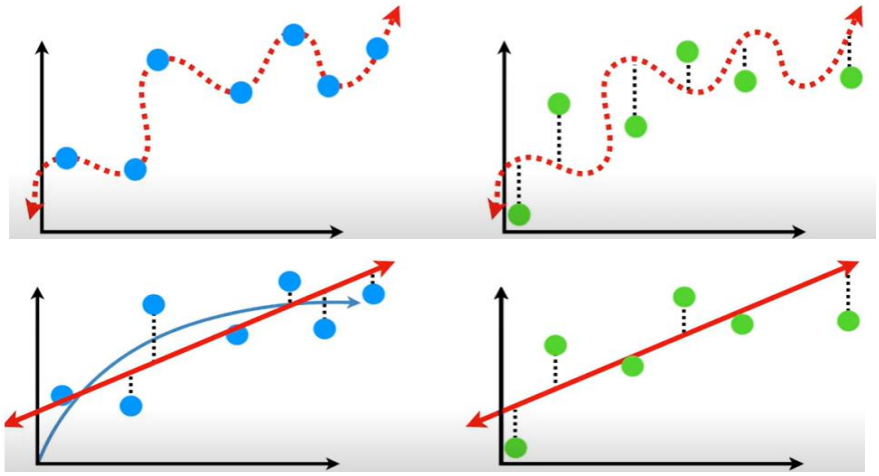


اما اگر بر فرض از mean squared error به عنوان cost function در اینجا استفاده می کردیم آنگاه نتیجه یک شکل non convex می شد که احتمال توقف گرادیانت دیسنت در هر local minimum آن وجود می داشت.



سوال 2-

وقتی تفاوت ارور دیتاست آموزشی با ارور دیتاست آزمایش شده زیاد باشد ، اصطلاحا مدل overfit بوده. این وقتی اتفاق می افتد که درجه چند جمله ای را به قدری بالا برده باشیم تا به حد زیادی مدل با داده های آموزشی فیت شود. و این یعنی از کلی بودن مدل که بتواند برای سایر داده ها هم ارور کمی داشته باشد کاسته شده پس انتظار می رود ارور دیتای آموزشی بسیار کم ولی دیتای آزمایشی بسیار بالا باشد.

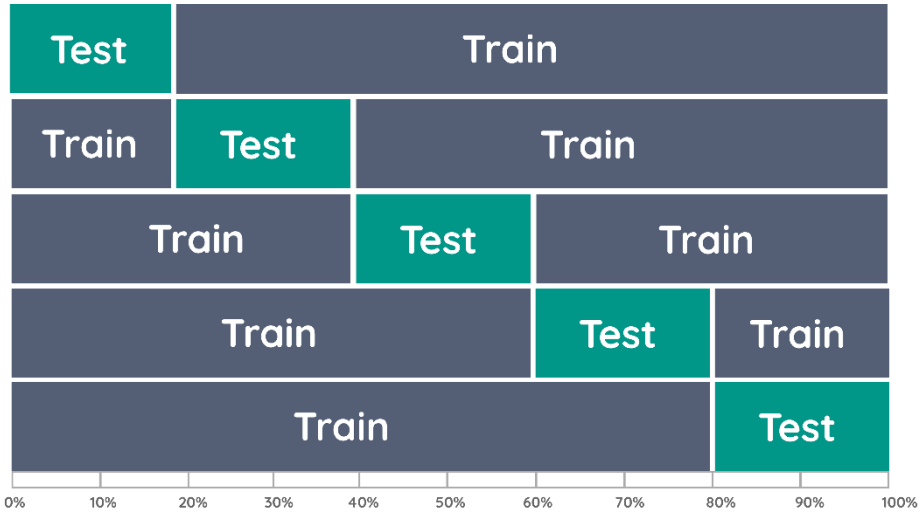


در شکل مقابل مشاهده می کنید که استفاده از مدلی بسیار پیچیده که کاملا فیت دیتای آموزشی باشد برای یک دیتاست متفاوت ارور بالایی خواهد داشت اما استفاده از مدل ساده تر جامع برای هر دیتاستی میزان معقول ارور خواهد داشت.

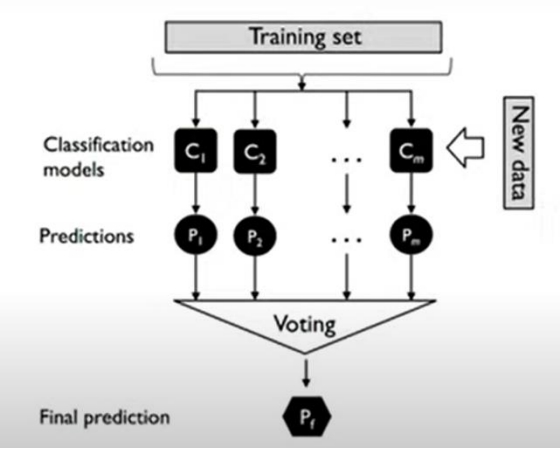
روش هایی برای اصطلاح یا پرهیز از overfit موجود می باشد که سه روش از آن ها به شرح زیر است:

الف) cross validation : در این روش به جهت پرهیز از اورفیت ، دیتاست آموزشی را به k بخش مساوی تقسیم می کنیم و یکی از بخش ها را به عنوان دیتاست آزمایشی انتخاب می کنیم و پس از ترین مدل با باقی گروه های دیتاست میزان ارور مدل برای این دسته آزمایشی را محاسبه می کنیم. این محاسبه را با جدا کردن هر یک از دسته ها و سپس میانگین گرفتن از ارور آن ها ادامه می دهیم تا اگر میزان ارور بالا بود به اصلاح مدل بپردازیم.

در این شکل نمونه ای از نحوه تقسیم بندی دیتاست آموزشی را مشاهده می کنید.



ب) ensemble modeling : به طور کلی به معنای این است که اگر چند مدل داشته باشیم ترکیب آن ها مدل قوی تری را می سازد که کمتر محتمل overfit است. به این ترتیب که خروجی تمام مدل ها را میانگین می گیریم.



پ) Regularization :

سه نوع روش متداول برای آن وجود دارد که آن ها را در زیر مشاهده می کنید .(L2 : ridge regression و L1 : lasso regression و elastic nest که ترکیب این دو است) می باشد . در این سه حالت با اضافه کردن مقدار پنالتی ای به cost function مقدار overfit کم شده به این ترتیب که برای کم کردن هزینه به طور کلی علاوه بر کم کردن مقدار قبل باید مقدار جدید را هم کم کند و به این ترتیب wi های کوچکتری انتخاب می شوند تا مدل بیشتر به سمت خط ساده تر برود.

$$\mathbf{L_2} \quad E(w) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, w) - t_n\}^2 + \lambda ||w||^2$$

$$||w||^2 = \mathbf{w}^T\mathbf{w} = w_0^2 + w_1^2 + \ldots + w_M^2$$

$$\mathbf{L_1} \quad E(w) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, w) - t_n\}^2 + \lambda ||w||_1$$

$$\textbf{Elastic net} \quad E(w) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, w) - t_n\}^2 + \lambda_1 ||w||_1 + \lambda_2 ||w||^2$$

لامبدا در این فرمول مقدار مثبت کوچکی بین 0 تا 1 است.

سوال 3-

به طور کلی واریانس بالا نشان دهنده overfit بودن مدل است و bias بالا نشان دهنده underfit بودن. وقتی میزان ارور دیتاست آموزشی و آزمایشی هردو زیاد باشد یعنی مدل ما بیش از اندازه ساده بوده و نیاز است پیچیده تر شود که این به معنای underfit بودن و در نتیجه داشتن bias بالا می باشد. چون مدل ridge regression است پس از ridge

regularization برای کاهش اورفیت استفاده شده اما مقدار لامبدا در آن زیاد بوده پس با کم کردن لامبدا مدل را به حالت پیچیده تر نزدیک تر می کنیم.

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
w_0^*	0.35	0.35	0.13
w_1^*	232.37	4.74	-0.05
w_2^*	-5321.83	-0.77	-0.06
w_3^*	48568.31	-31.97	-0.05
w_4^*	-231639.30	-3.89	-0.03
w_5^*	640042.26	55.28	-0.02
w_6^*	-1061800.52	41.32	-0.01
w_7^*	1042400.18	-45.95	-0.00
w_8^*	-557682.99	-91.53	0.00
w_9^*	125201.43	72.68	0.01

همانطور که در جدول مشاهده می کنید. مقدار بزرگ تر لامبدا به معنای کوچکتر بودن wi ها و ساده تر بودن مدل است.

سوال 4-

الف) وقتی مدل اورفیت شده باشد یعنی cost function برای دیتاست آموزشی بسیار ایده آل باشد محتمل است این مدل نتواند برای دیتاست پیشبینی معقولی بکند. پس از ridge regression به جهت افقی تر کردن مدل برای مدل سازی بهتر استفاده می کنیم (انتخاب خطی با شیب کمتر) تا به نقطه تعادل برسیم که مجموع rss و پنالتی ridge regression کمینه باشد.

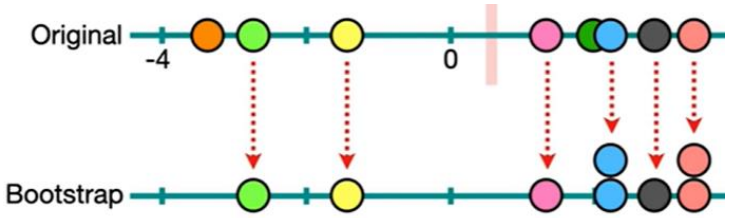
ب) به طور کلی اگر تعداد wi ها کم باشد و تاثیر آن ها زیادتر بهتر است از lasso بجای ridge استفاده شود . این حالت مشمول حالتی هم می شود که تعداد زیادی wi داریم و می دانیم برخی از آن ها تقریبا بی اثر هستند و فقط برخی هستند که تاثیر بر روی مدل دارند.

پ) وقتی که فیچر های زیادی داشته باشیم و هم بستگی بین چندین فیچر وجود داشته باشد elastic net انتخاب بهتری از هردو ridge regression و lasso regression می باشد . زیرا در حالی که در این حالت lasso از بین فیچر هایی که با هم دچار هم بستگی شدند فقط یکی را انتخاب می کند و سایر را حذف می کند و ridge تمام آن ها را تاثیر می دهد ، elastic net فقط برخی از آن ها را نگه داشته و سایر را حذف می کند. در حالت کلی elastic net به عنوان حالت میانه ای بین ridge و lasso عمل می کند.

سوال 7_

فرضا وقتی می خواهیم تعداد داده بیشتری داشته باشیم بجای تکرار چند بار آزمایش می توانیم از روش کم هزینه تر bootstrapping استفاده کنیم. به معنای انتخاب داده از دیتاست (sampling with replacement) می باشد. یعنی چندین مرتبه به اندازه دیتاست داده انتخاب می کنیم . از این دیتاست های جدید بدست آمده می توان برای تخمین statistic های مختلف استفاده کرد مانند میانگین ، میانه یا ... . اما cross validation که در بالا به توضیح آن پرداختیم روشی می باشد

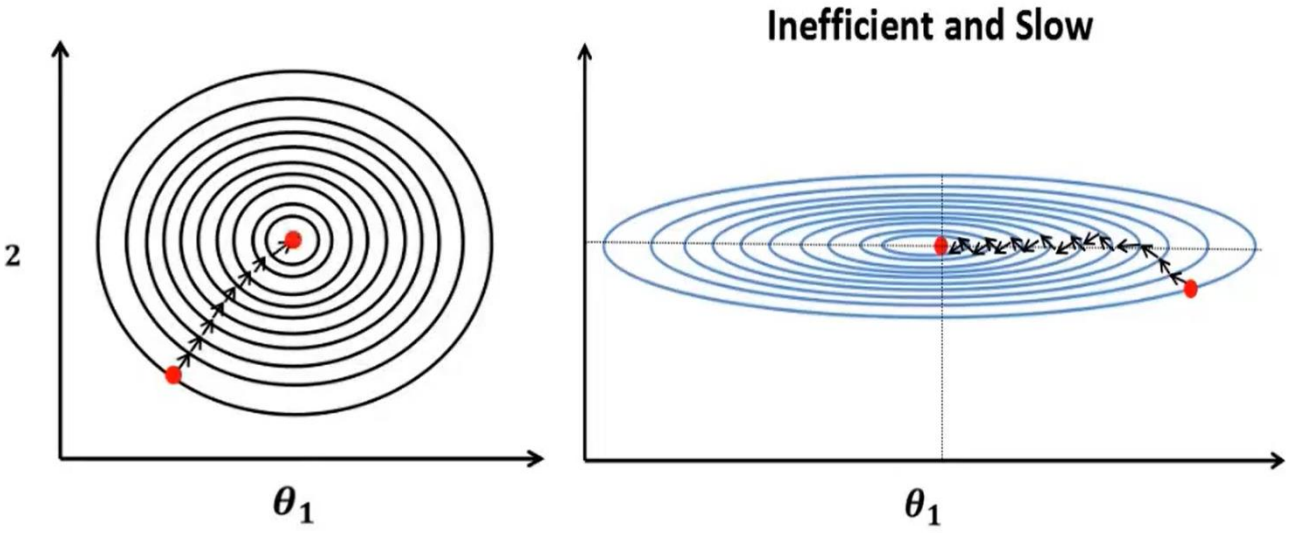
که برای افزایش دقت یا غلبه بر اورفیت استفاده می شود به این ترتیب که دیتا را پارتیشن بندی می کند و چندین بار خروجی می گیرد و میانگین آن هارا به عنوان خروجی نهایی باز می گرداند. برای این که تشخیص بدهیم از کدام تکنیک برای افزایش بازدهی مدل استفاده کنیم بستگی به پیچیدگی و سایز دیتاست دارد اما در حالت کلی با توجه به ویژگی های بیان شده از آن ها وقتی سایز دیتاست کوچک است bootstrapping کاربرد بیشتری دارد و در سایز بالاتر cross validation.



نمونه ای از انتخاب نمونه bootstrap شده از دیتاست

سوال 11_

در گرادیانت دیسنت زیرا که یک الگوریتم iterative می باشد . با اسکیل نبودن فیچر ها سرعت کارایی آن کاهش می یابد به این ترتیب که برای یافتن نقطه global minimum مسیر طولانی تری لا می پیماید.



در شکل سمت راست گرادیانت دیسنت بر روی فیچر های اسکیل نشده را می بینید و شکل سمت چت پس از اسکیل شدن فیچر ها می باشد.

در normal equation چون هیچ iterate ای اتفاق نمی افتد و صرفا حل یک معادله است معمولا نیاز به فیچر اسکیلینگ نمی باشد مگر اینکه در معادله آن قسمت inverse کردن از روشی استفاده شود که با فیچر های اسکیل شده به بازدهی بهتری برسیم.

در مورد svd هم به طور کلی تاثیرگذاری ندارد زیرا svd یک روش جبر خطی می باشد اما در حالت های خاص که اسکیل برخی فیچر ها خیلی بالا باشد امکان دچار شدن به ارور های غیرجزئی می باشد.

برای حل این مشکل از feature scaling استفاده می شود که دیتای تمام فیچر هارا در یک رنج قرار بدهد. دو روش رایج آن min-max normalization می باشد که برای اسکیل فیچر ها بین صفر تا یک است د روش دیگر standard normalization که فیچر هارا به فرم استاندارد شده نشان می دهد.

$$X_{new} = \frac{X_i - \min(X)}{\max(x) - \min(X)}$$

Min-max normalization

$$x_{new} = \frac{x - \mu}{\sigma}$$

Standard normalization