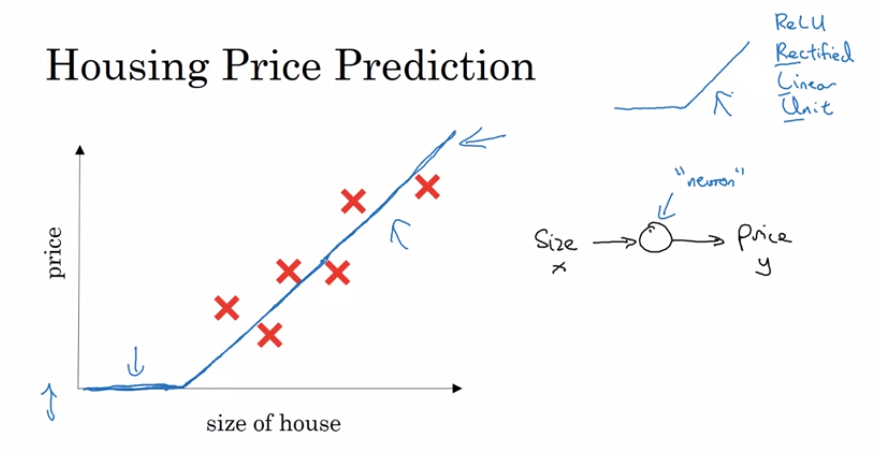
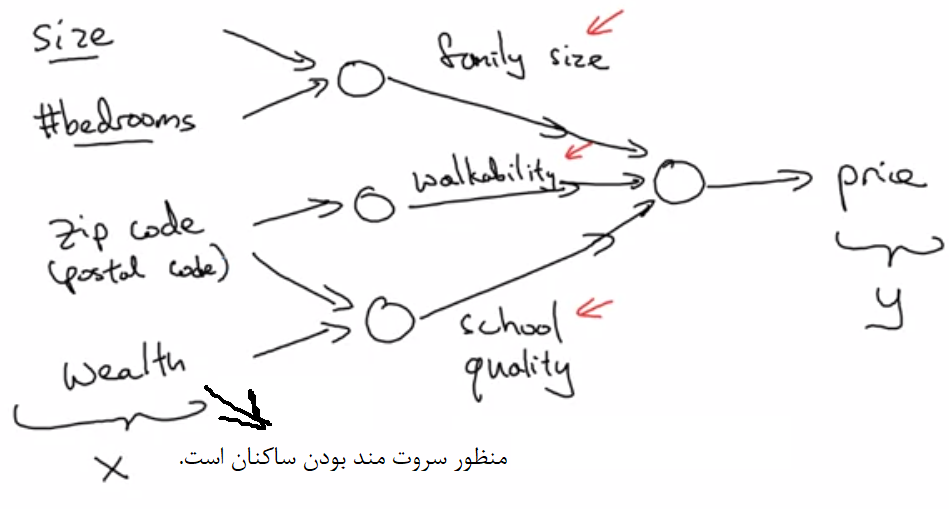
Machin Learning

**Neural Networks and Deep Learning:**

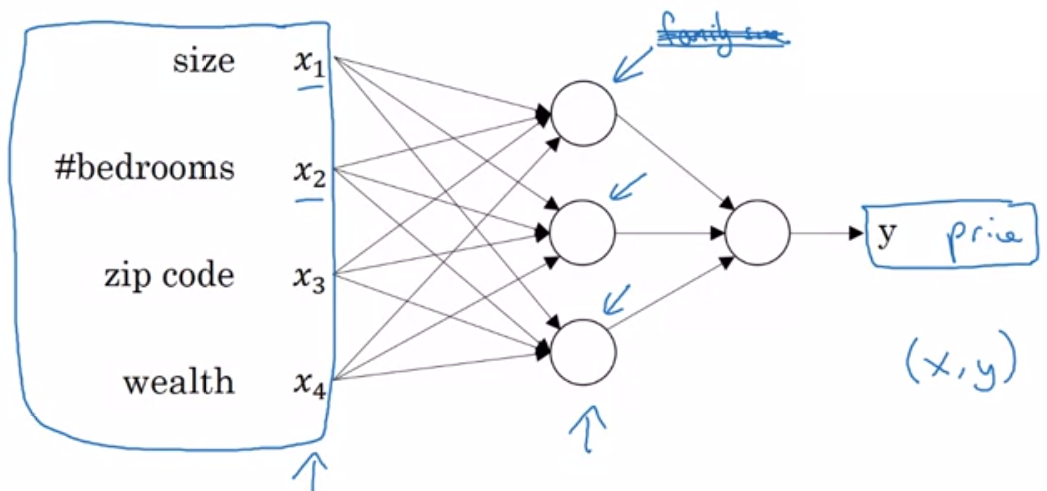
1. زمانی که صحبت از machine learning می‌شود منظور همان شبکه‌های عصبی است.
2. یک مثال از شبکه‌های عصبی را در تصویر زیر مشاهده می‌کنید. که در آن از یک عصب یا neuron برای محاسبه یا پیش بینی مقدار قیمت یک خانه بر اساس سایزی که به neuron داده می‌شود انجام می‌شود. ReLU (یا Rectified Linear Unit) در واقع به linear regressionی گفته می‌شود که از صفر شروع شده و صعودی می‌شود(در واقع کلمه Rectified به معنی اصلاح شده نیز به همین منظور استفاده شده است).



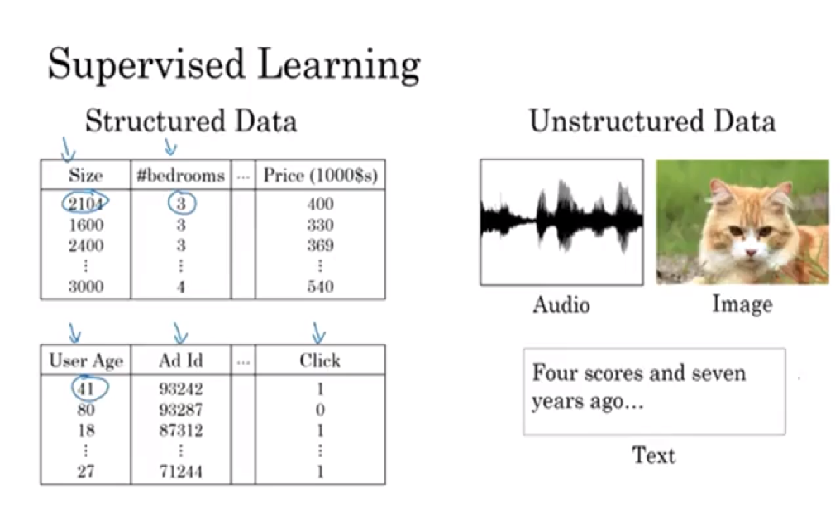
1. در مثال بالا یک شبکه عصبی بسیار کوچک را مشاهده کردیم. اگر برای حدس زدن قیمت خانه از علاوه بر سایز از ویژگی(feature)های دیگری نیز استفاده کنیم neuronهای بیشتری نیاز داریم و یک شبکه بزرگتر خواهیم داشت:



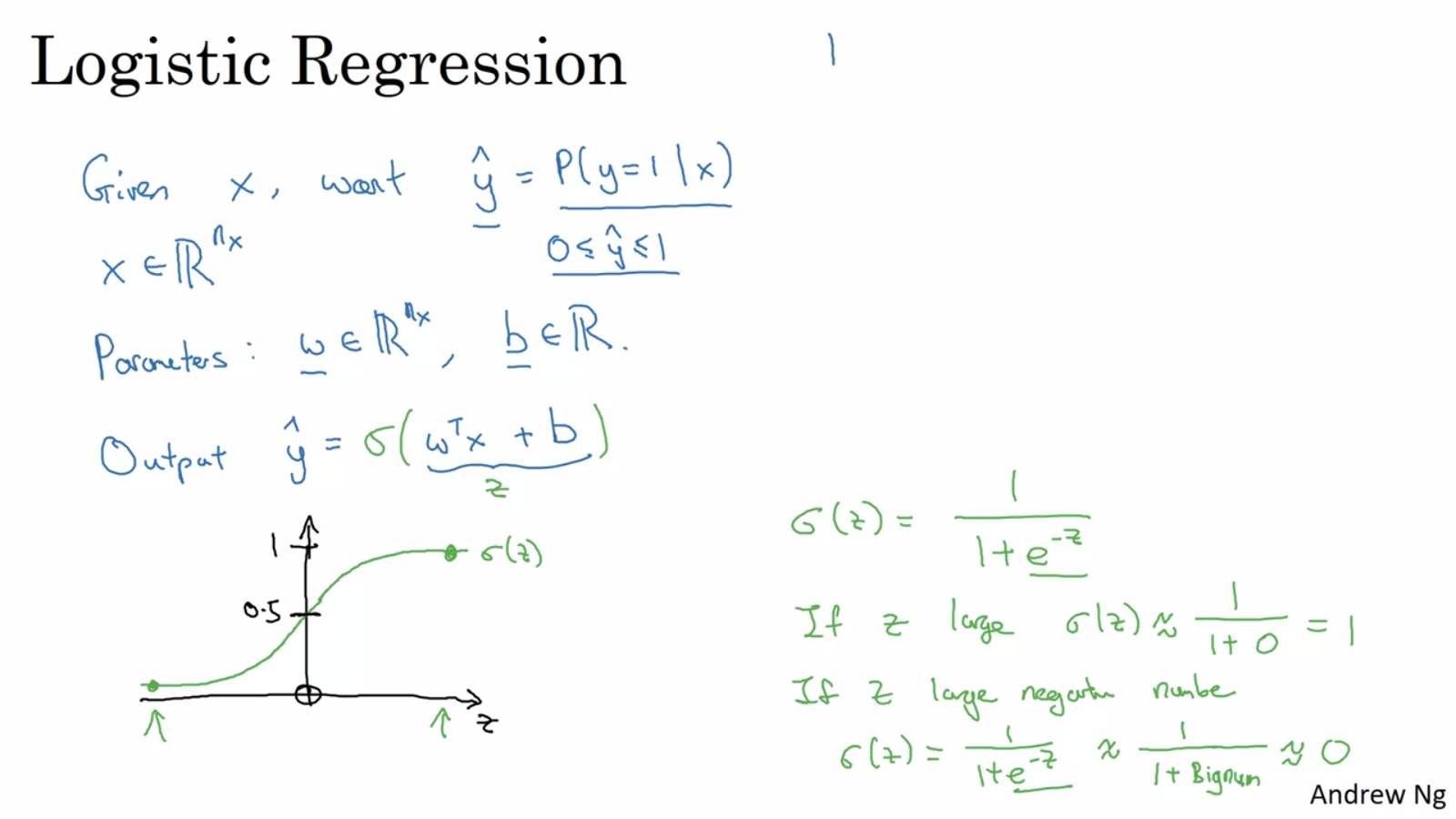
1. ولی ما در machine learning شبیه به شکل زیر شبکه خود را تشکیل می‌دهیم. درواقع همه ورودی‌ها به همه neuronها وارد می‌شوند.



1. به روشی که به ماشین تعداد زیادی داده به همراه نتیجه داده می‌شود برای مثال قیمت خانه بر اساس سایز و ... و از ماشین می‌خواهیم که حالا با یک مقدار ورودی قیمت خانه را حدس بزند supervised learning گفته می‌شود.
2. در supervised learning داده ممکن است به دو صورت structured و یا unstructured باشد. در structured داده‌ها دارای مفهوم و قابل فهم هستند.



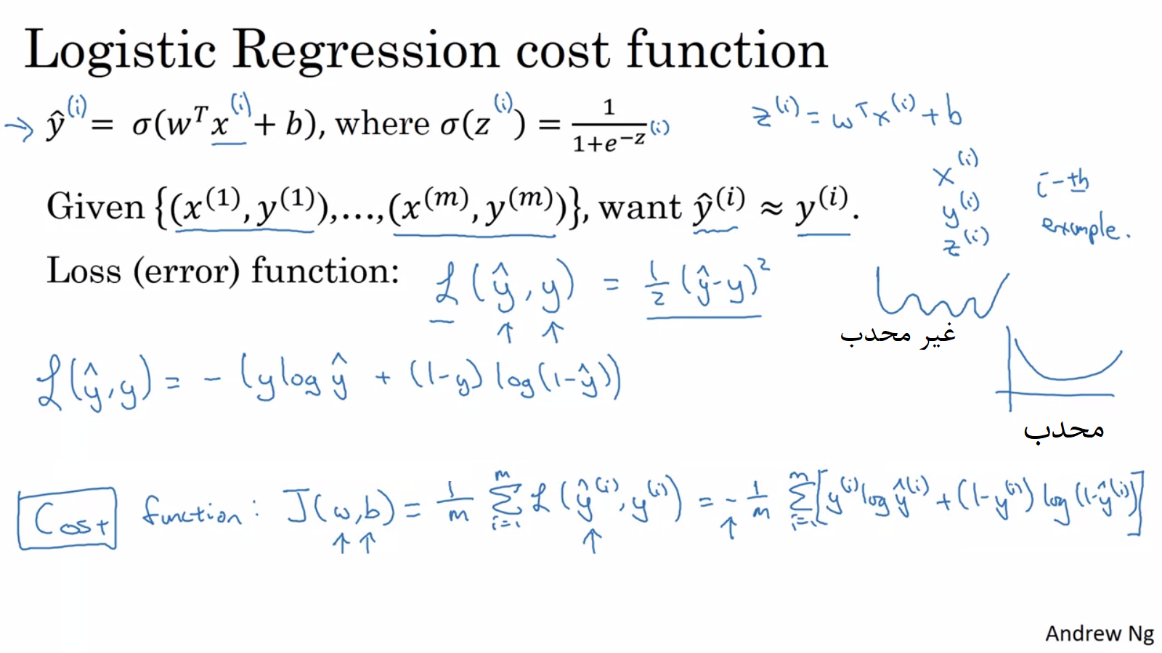
1. Logistic regression یک الگوریتم برای انجام binary classification است. برای مثال با دادن یک تصویر به الگوریتم انتظار داشته باشیم خروجی به ما بگوید که تصویر داده شده یک گربه است یا خیر. در واقع به ازای ورودی خروجی فقط می‌تواند دو حالت داشته باشد.
2. برای راحتی در ادامه بحث در اینجا. برخی از Notationهایی که در این در خواهیم دید را معرفی خواهیم کرد:
   * (x,y) نشان دهنده ورودی x وخروجی y است. که در آن و برای logistic regression می‌باشد. m = {(x1, y1), … , (xm,ym)} نشان دهنده training set می‌باشد.
3. در Logistic regression کار به این صورت خواهد بود که با دادن یک مقدار x به سیستم انتظار خروجی را داریم که در آن تخمین احتمال اینکه به ازای ورودی x مقدرا باشد است. برای پیدا کردن به تابعی نیاز داریم که با دادن ورودی به آن مقداری بین صفر و یک ایجاد کند. به این منظور از تابع sigmoid (به اختصار ) استفاده می‌کنیم. به تصویر زیر نگاه کنید. درواقع همان فرمول خط است(برای آشنایی با پارامترهای فرمول خط به شماره بعد مراجعه کنید. قصد ما این است که با توجه به مقادیر (x,y) که به عنوان داده train دریافت می‌کنیم مقدار w و b به گونه‌ای پیدا کنیم که مقدار تخمین ما به مقدار واقعی نزدیک شود. به عبارت دیگر به دریافت یک مقدار x جدید و جاگزاری در فرمول مقدار تخمینی از y به دست آوریم.



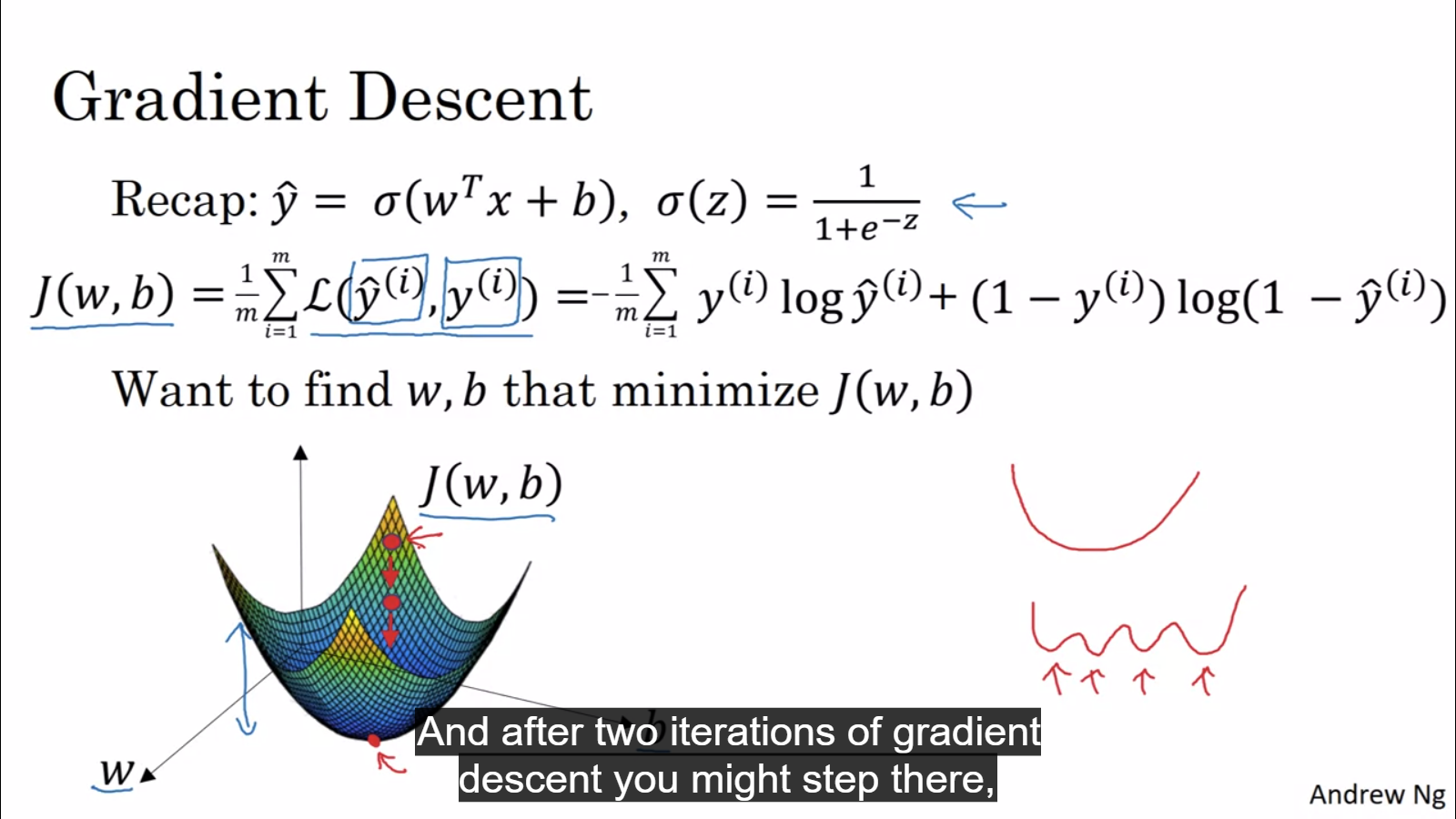
1. فرمول خط روی نمودار:



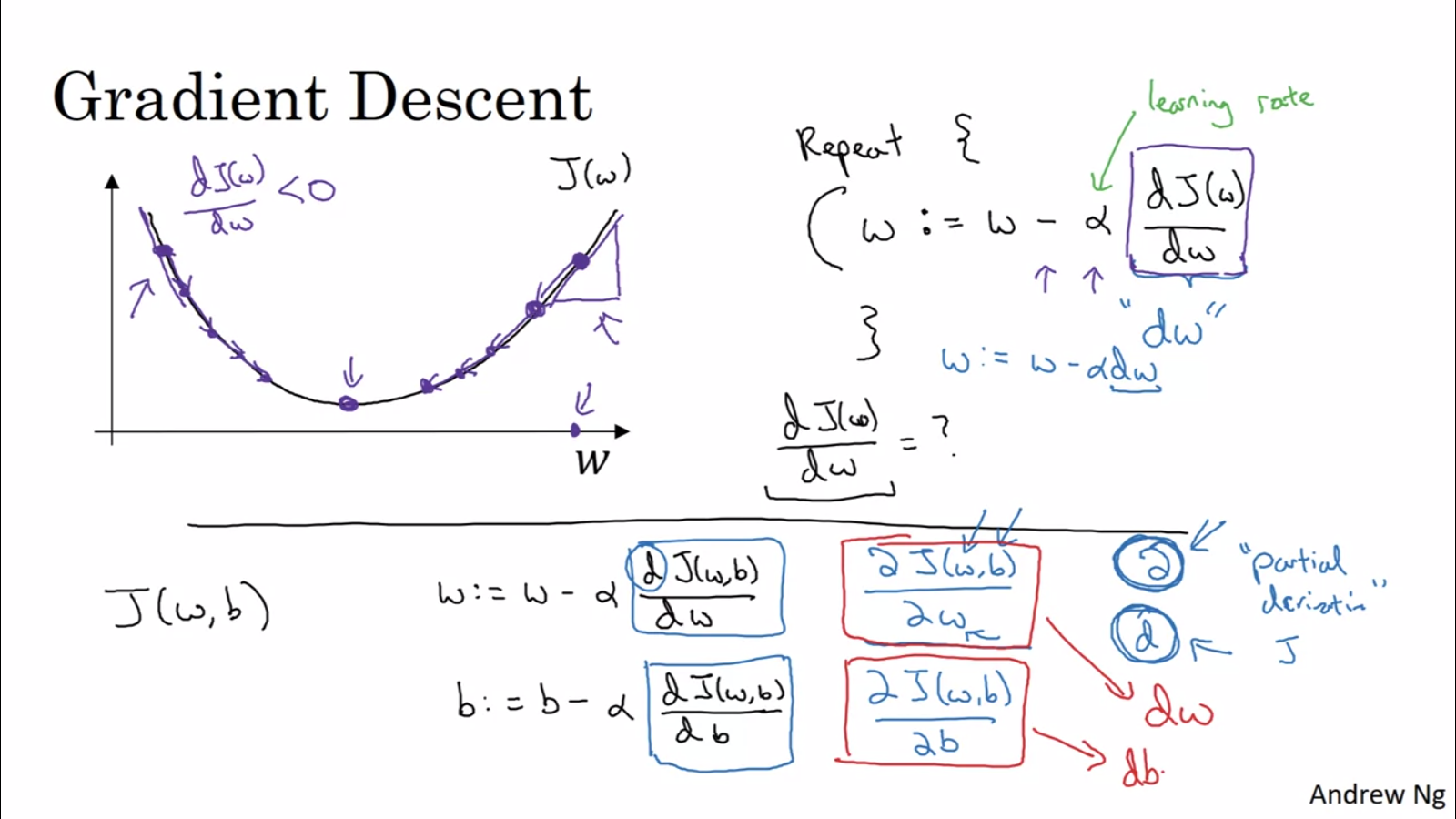
1. در تصویر زیر برای loss function دو تابع معرفی شده است. که ما از تابع دوم استفاده خواهیم کرد و دلیل آن این است که نمودار فرمول محدب نمی‌شود و بعداً برای optimization مشکل خواهیم داشت. Cost function در واقع میانگین loss function برای همه test caseها است. به عبارت دیگر loss function مقدار خطا را برای یک test case اندازه میگیرد ولی cost function نشان می‌دهد که ما چه مقدار مقدار w و b را برای که test caseها درست حدس زده ایم (توجه شود که پایه لوگاریتم e است).



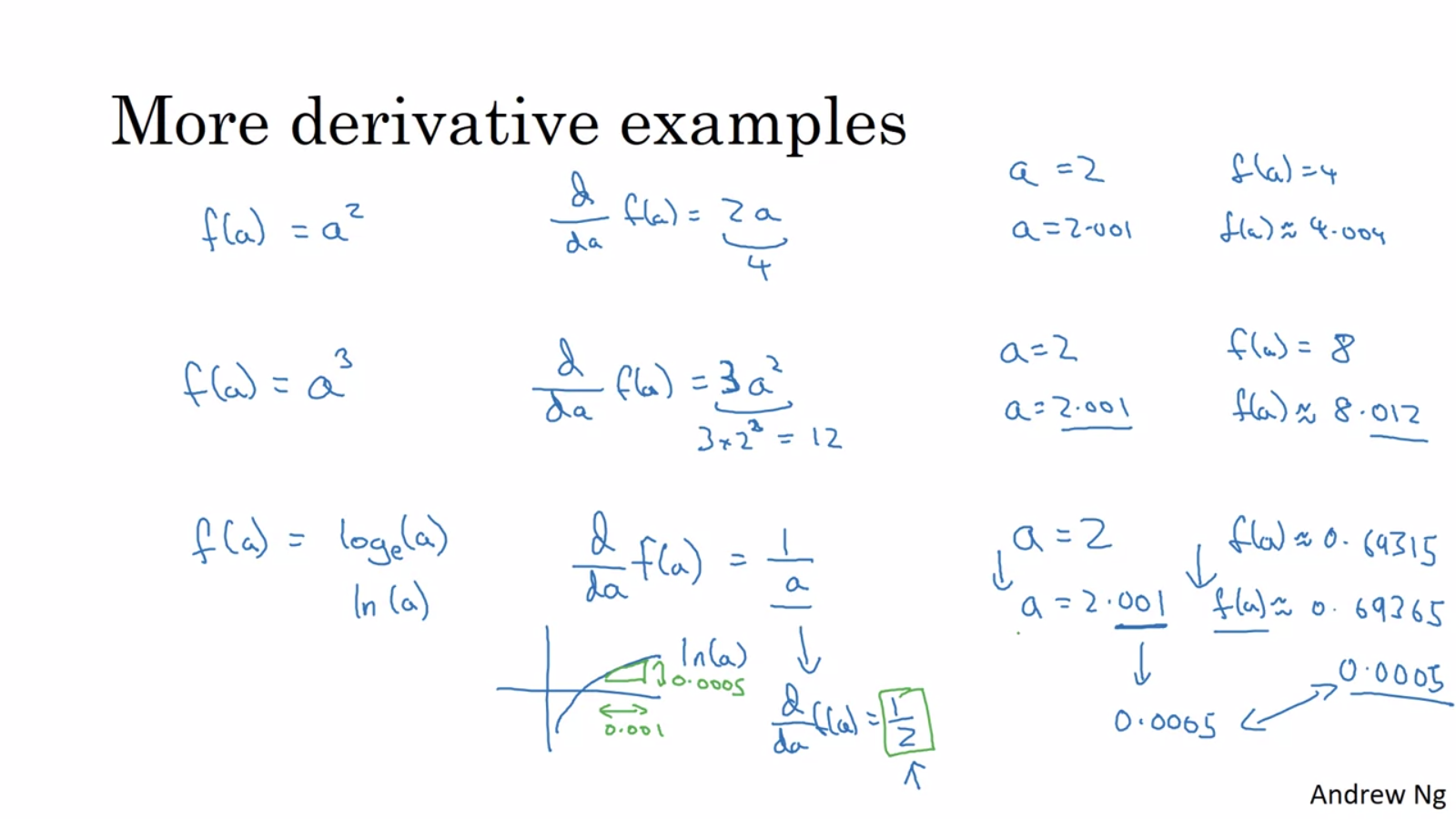
1. تا اینجا learning function و cost function را یادگرفتیم. حال باید به گونه‌ای مقدار w و b را در learning function پیدا کنیم که مقدار cost function کمترین مقدار خود را داشته باشد. در تصویر زیر نمودار مقدار cost function بر اساس w و b رسم شده است(البته w و b می‌توانند چند بعدی باشند ولی برای راحتی و قابل ترسیم بودن تک بعدی فرض شده‌اند). هدف این است که در نمودار مقدار w و b را که در کف نمودار قرار دارد پیدا کنیم. برای این کار از تابع gradient descent استفاده کنیم. به این صورت که با مقدار دهی اولیه w و b این تابع با انتخاب w و b بهتر در هر مرحله به سمت کف نمودار حرکت کرده و درنتیجه کمترین مقدار cost function پیدا شود در آن نقطه converge کند. مقدار دهی اولیه در logistic regression مقدار صفر برای هر دو پارامتر w و b است. توجه شود که ما cost function یا J(w,b) را به گونه‌ای انتخاب کردیم که نمودار آن convex باشد.



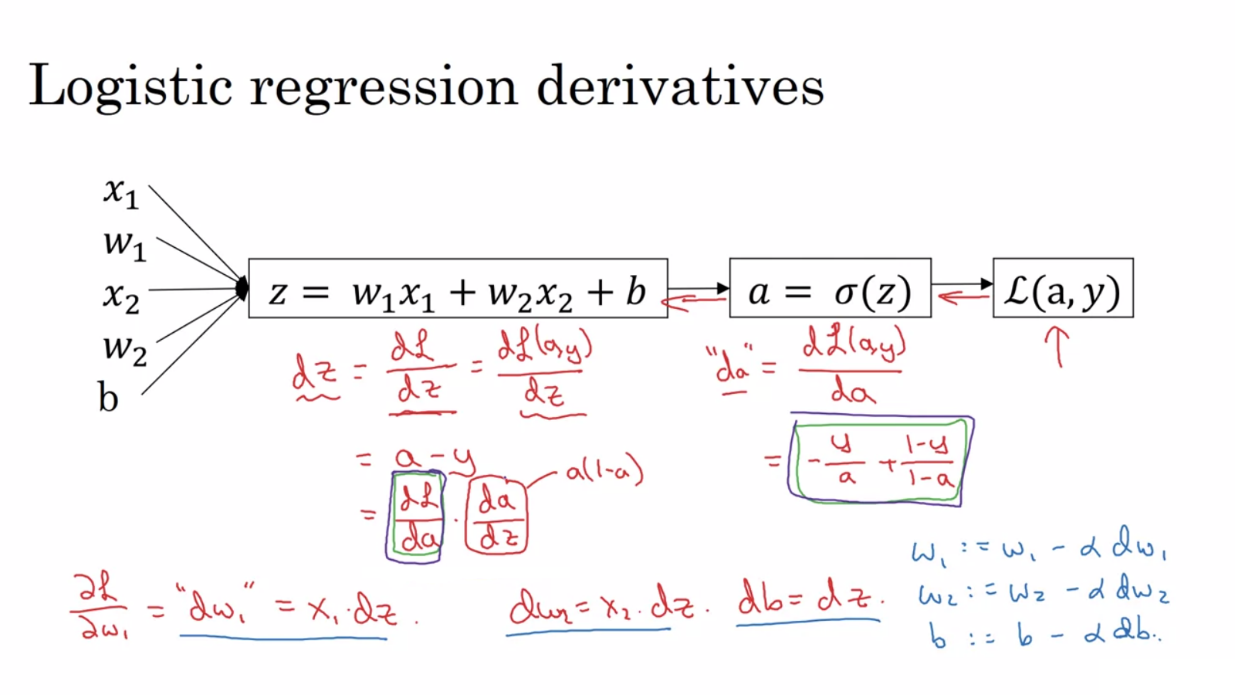
1. در تصویر زیر قسمت پایین تصویر نحوه محاسبه gradient descent را برای دو مقدار w و b مشاهده می‌کنید که در هر مرحله برای w مقدار را از w کم می‌کنیم. مشتق است(مشتق در واقع شیب را محاسبه می‌کند و در واقع مقداریست که با آن مشخص می‌کنیم در هر مرحله چه مقدار پرش انجام شود تا سریع تر به مقدار مطلوب برای w دستیابیم.

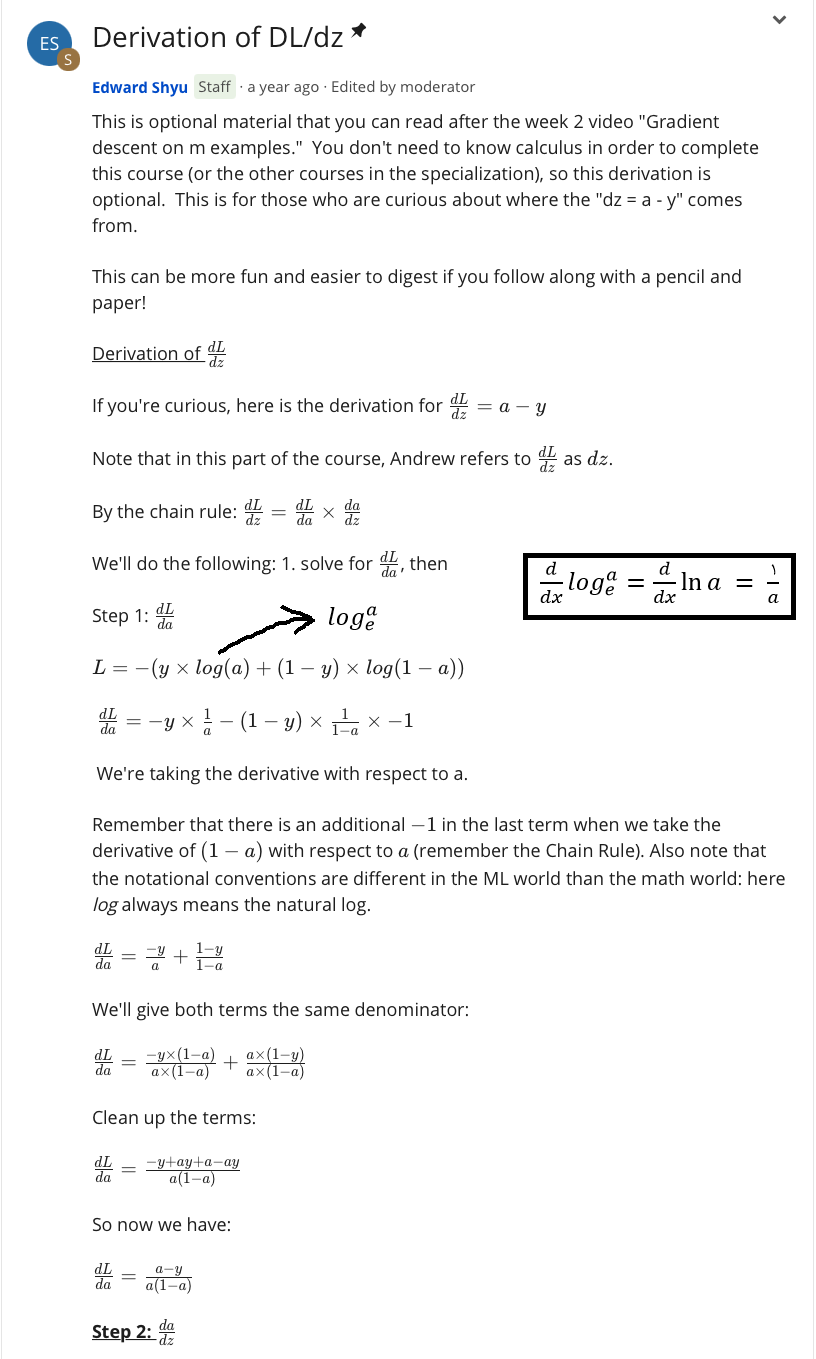


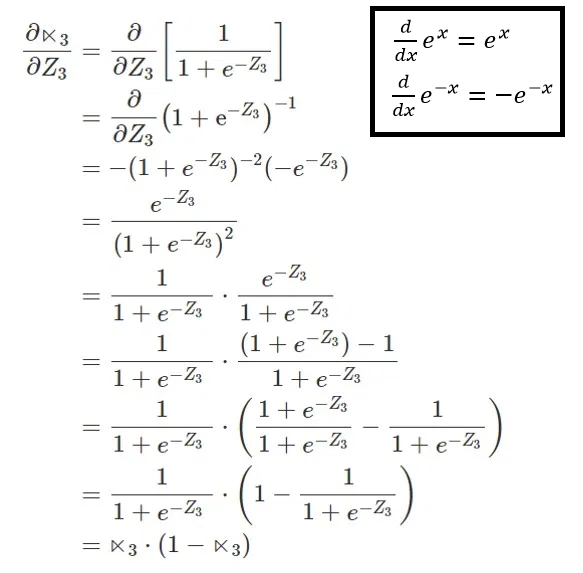
1. چند مثال از مشتق:

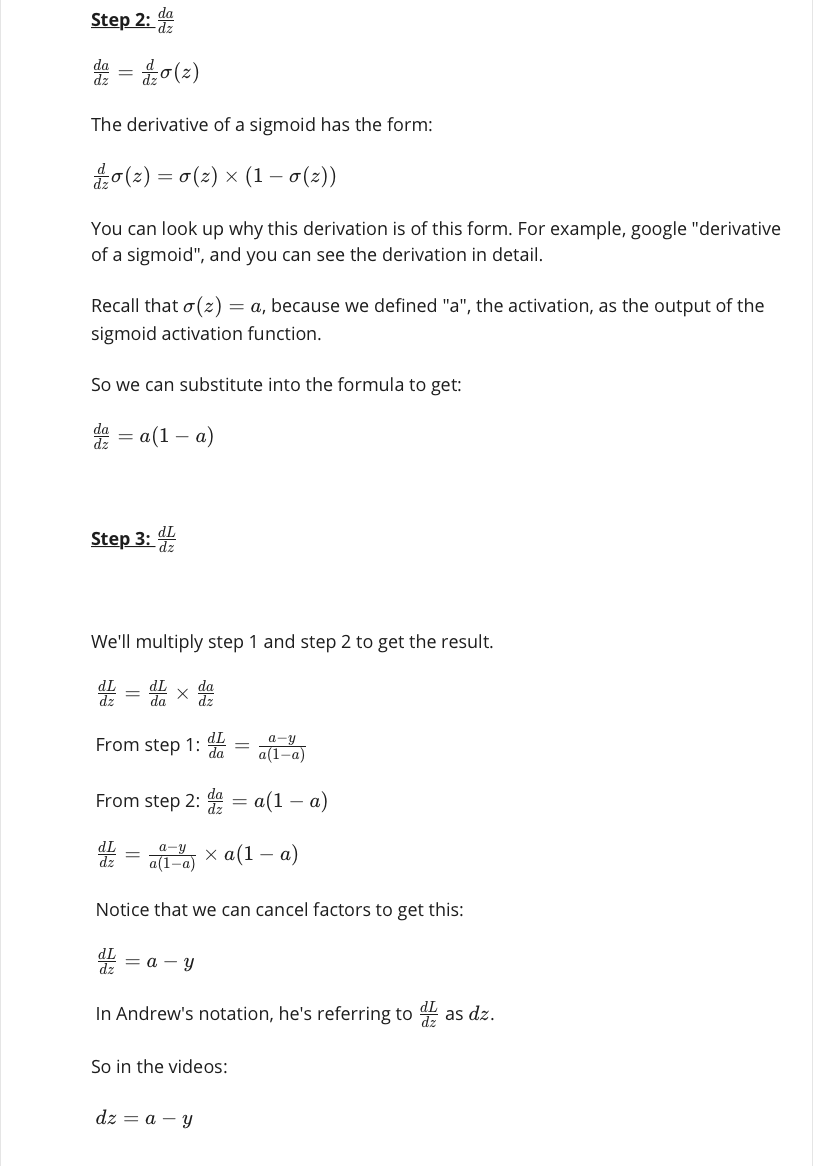


1. همان طور که در دو شماره قبل گفته شده برای پیدا کردن بهترین مقادیر w و b باید از gradient descent استفاده کنیم. و برای محاسبه gradient descent هرکدام از مقادیر w و b نیاز به مشتق آنها بر اساس cost function داشتیم (مثلاً ). در در تصویر زیر مشتق را برای تنها یک test case مشاهده می‌کنید نه برای cost function که در قسمت های بعدی درباره آن صحبت می‌کنیم. (توجه شود که نحوه محاسبه مشتق‌ها در شماره‌های بعدی قرار دارد)

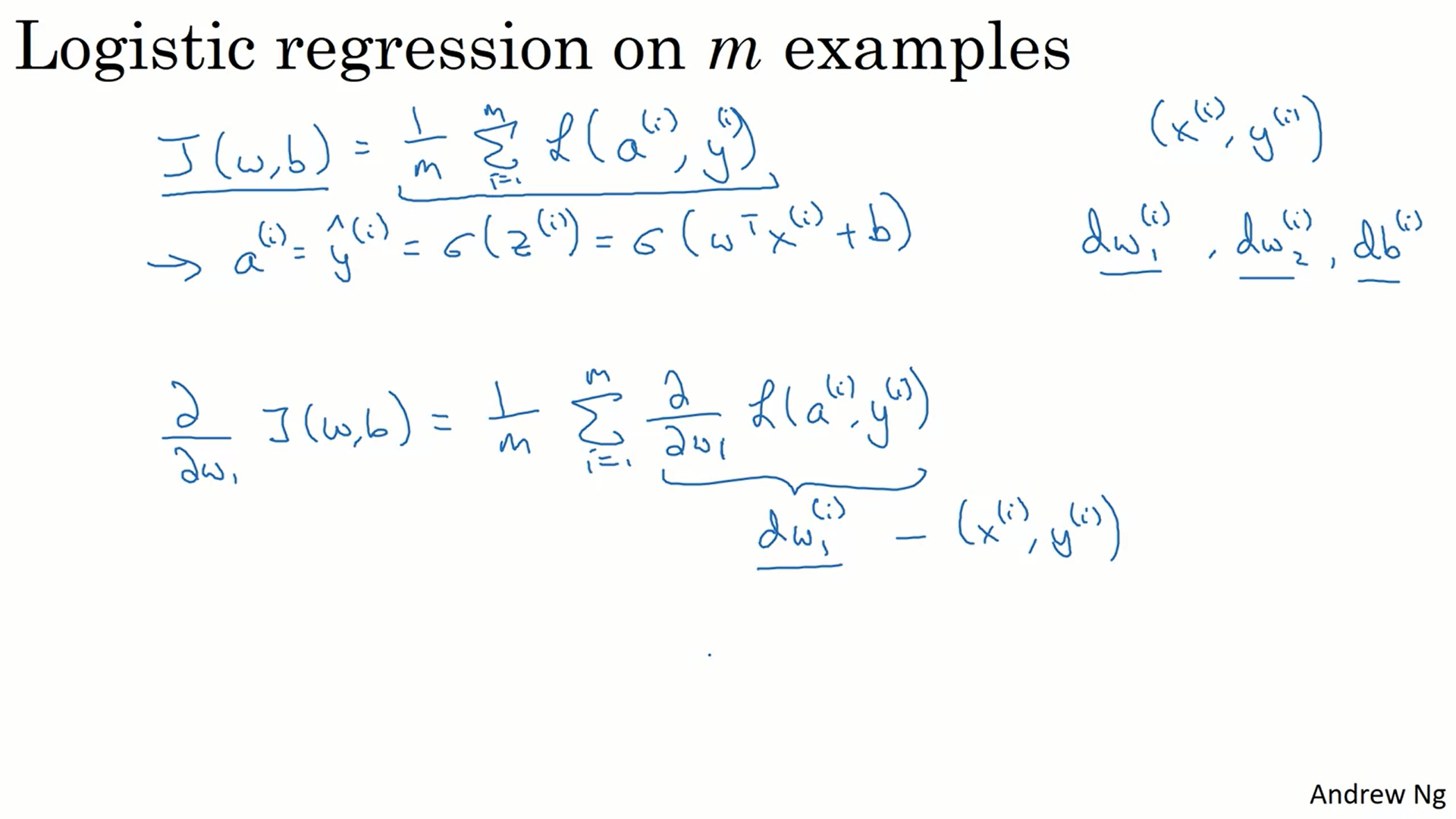




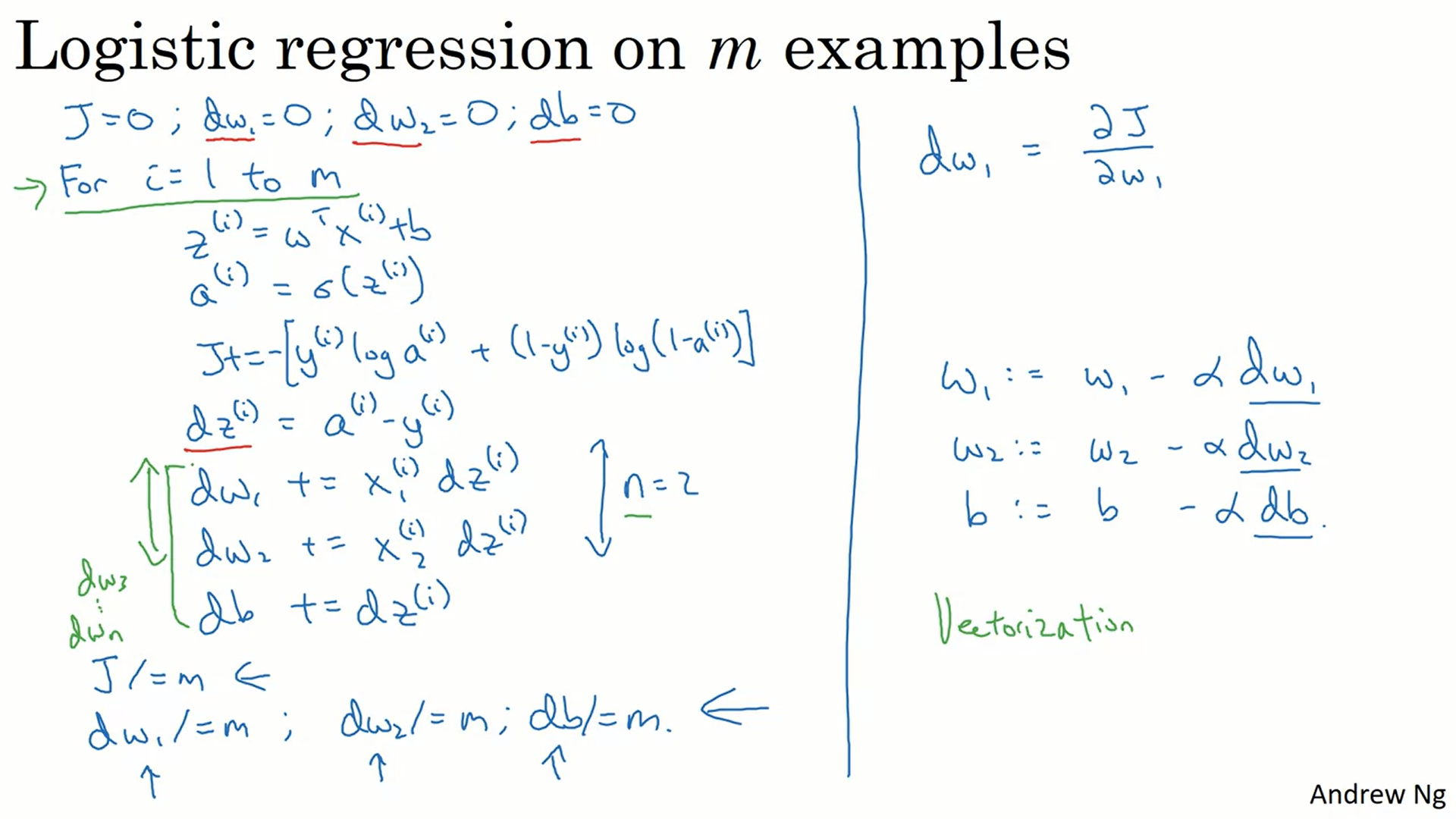




1. تا به اینجا ما نحوه محاسبه gradient descent تنها برای یک test case را مشاهده کردیم. برای محاسبه آن برای cost function باید مشتق‌ها را بر m تقسیم کنیم



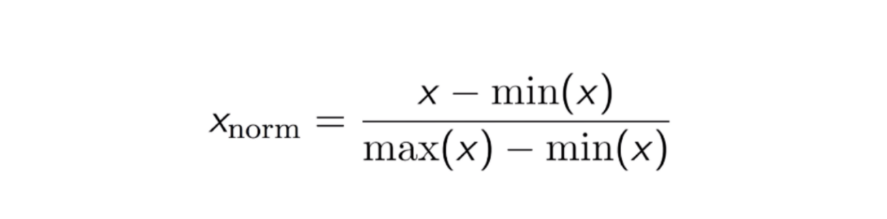
1. در شکل زیر یک شبه کد برای محاسبه gradient descent برای cost function را مشاهده می‌کنید. البته این کد کامل نیست و در آن فرض شده که دو feature یا دو x برای ورودی داریم.



1. برای محاسبه dw در کد بالا دو راه داریم یکی استفاده از for-loop و دیگری vectorization یا به عبارتی ایجاد وکتوری از x و dz و ضرب دکارتی آنها برای انجام vectorization زبان python امکانات و performance بسیار خوبی ارائه می‌کند و ما نیز از vectorization استفاده خواهیم کرد. کد زیر دو روش را باهم مقایسه کرده است:



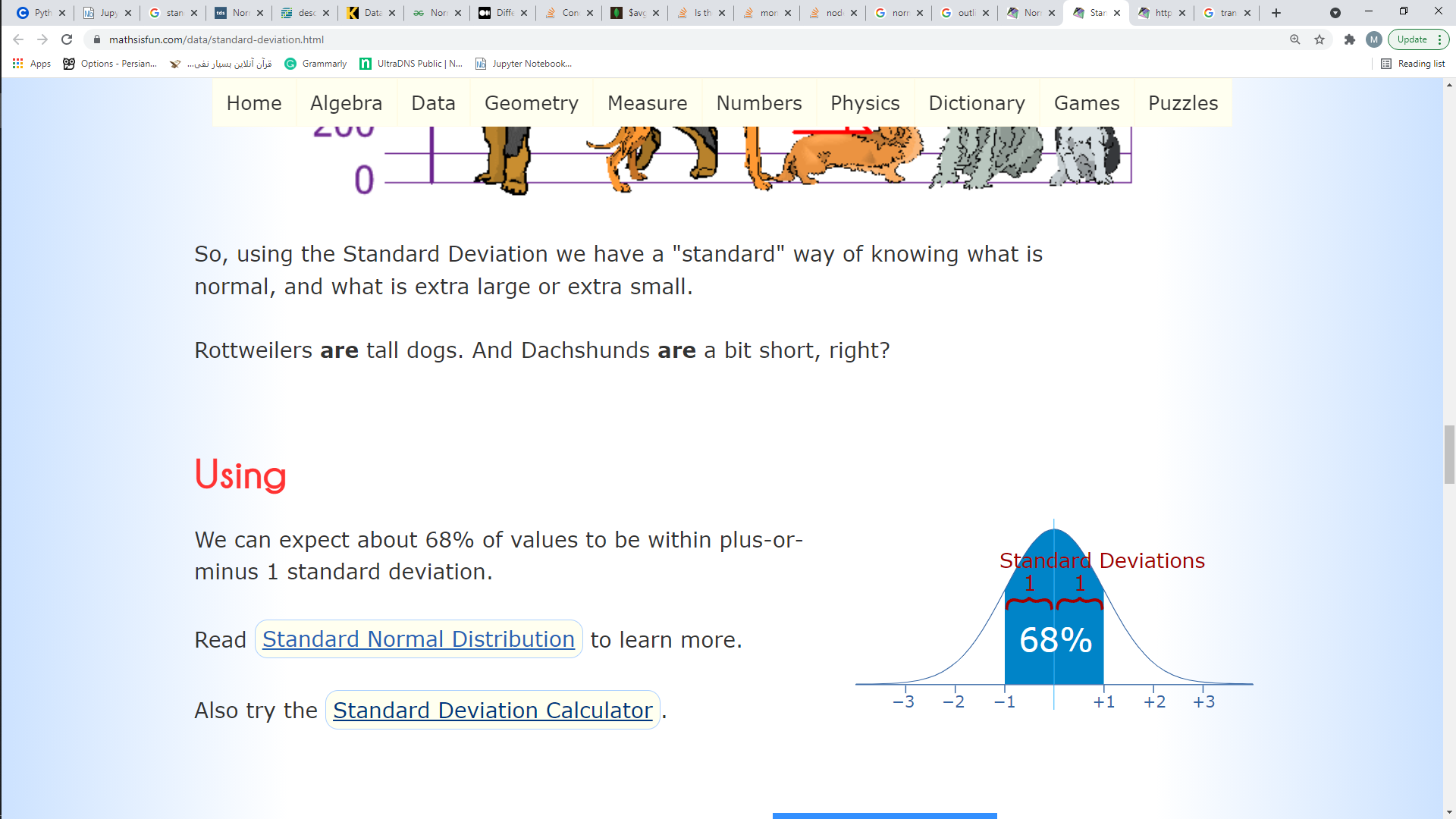
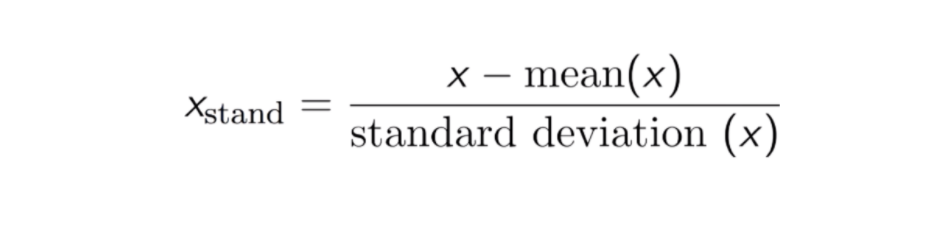
1. یک قانون کلی در برنامه نویسی شبکه‌های عصبی اجتناب استفاده از for-loop در مواقع ممکن است.
2. نکته: زمانی که مقیاس featureها، مختلف باشد یا به عبارتی فاصله آنها زیات باشد. برای مثال فرض کنید یک feature نمره باشد که مقدار آن بین 0 تا 10 است و یک feature متراژ خانه باشد که معمولاً بالای 150 است در این صورت مقادیر متراژ می‌تواند نتایج را به سمت خود بایاس کند. به عبارتی به این معنی باشد که متراژ تاثیر بیشتری در خروجی دارد درحالی که تاثیر هر دو feature یکسان است. در این شرایط برای هم مقیاس کردن داده‌ها را نرمال می‌کنیم. دوتا از فرمول‌های نرمال سازی به شکل زیر است. یکی دیگر از فواید نرمال سازی داده‌ها این است که بعد از نرمال سازی gradient descent سریع‌تر همگرا می‌شود.



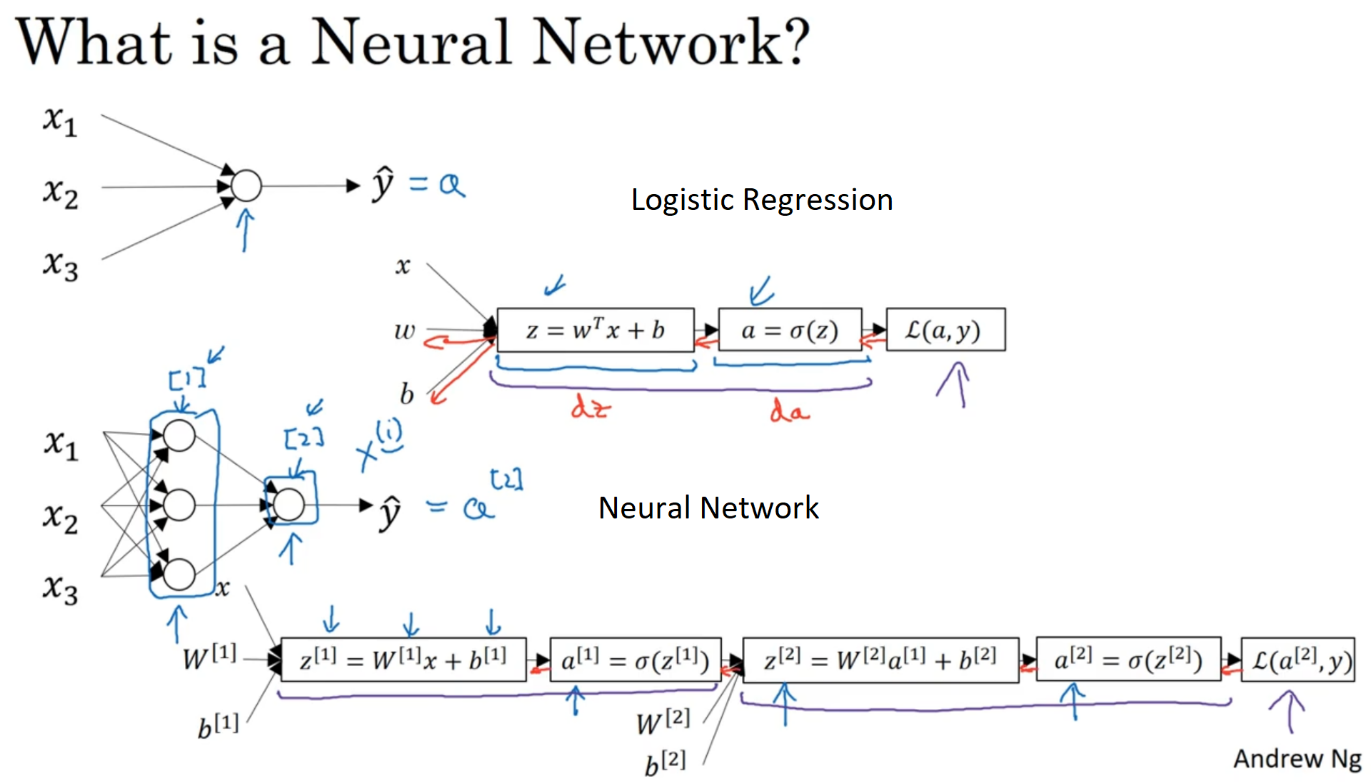
1. Normalization و standardization (روش استاندارد سازی در ادامه گفته شده) هر دو روش‌هایی برای rescale کردن داده‌ها هستند تا performance فرآیند یادگیری افزایش یابد. تفاوت normalization با standardization در:
   * در normalization داده‌ها در رنج بین 0 تا یک قرار می‌گیرند ولی در standardization رنج مشخصی وجود ندارد.
   * در standardization داده های outlier تاثیر زیادی ایجاد نمی‌کند. به عبارت دیگر فاصله داده‌ها بعد از standardization حفظ می‌شود و داده‌های پرت همچنان پرت هستند ولی در normalization داده‌ها پرت از دست می‌روند یا به عبارتی بعد از normal شدن قابل تشخیص نیستند.

به صورت کلی زمانی که توزیع داده‌ها normal و یا Gaussian است standardization بهتر از normalization است ولی اگر توزیع داده‌ها را نمیدانیم بهتر است از normalization استفاده شود.

1. Standardization: برای استاندارد سازی داده‌ها از فرمول زیر استفاده می‎‌کنیم که همان فرمول توزیع نرمال استاندارد است که به آن z-score نیز گفته می‌شود. در z-score میانه داده‌ها صفر است و انحراف معیار(یا standard deviation) یک است(به نمودار توجه شود). معنی انحراف معیار این است که 68 درصد داده‌ها یا نرمال آنها در فاصله 1± از میانه قرار دارند:



1. برای rescale کردن داده عکس یکی از روش‌ها مرسوم تقسیم مقادیر آن به 255 است.
2. ما تا به اینجا روش logistic regression را یاد گرفتیم. که در آن یک نورون داشتیم و با استفاده از backward propagation (به این معنی که با مشتق از loss function به مقادیر مناسب برای w و b دست میافتیم) کار می‌کردیم. در شبکه عصبی دارای لایه‌ها مختلفی که فعلاً با یک لایه کار می‌کنیم هستیم. که هر لایه دارای تعدادی nuron است کار داریم. هر nuron یا node در شبکه دارای یک activation function است که خروجی آن را به لایه بعدی تحویل می‌دهد:



1. یب
2. یب
3. سیب
4. سی