【機器學習】深入探討，為什麼要做特徵歸一化/標準化?

機器學習算法與Python實戰 今天

↑↑↑點擊上方

作者 | shine-lee

來源 | https://blog.csdn.net/blogshinelee/article/details/102875044

導讀

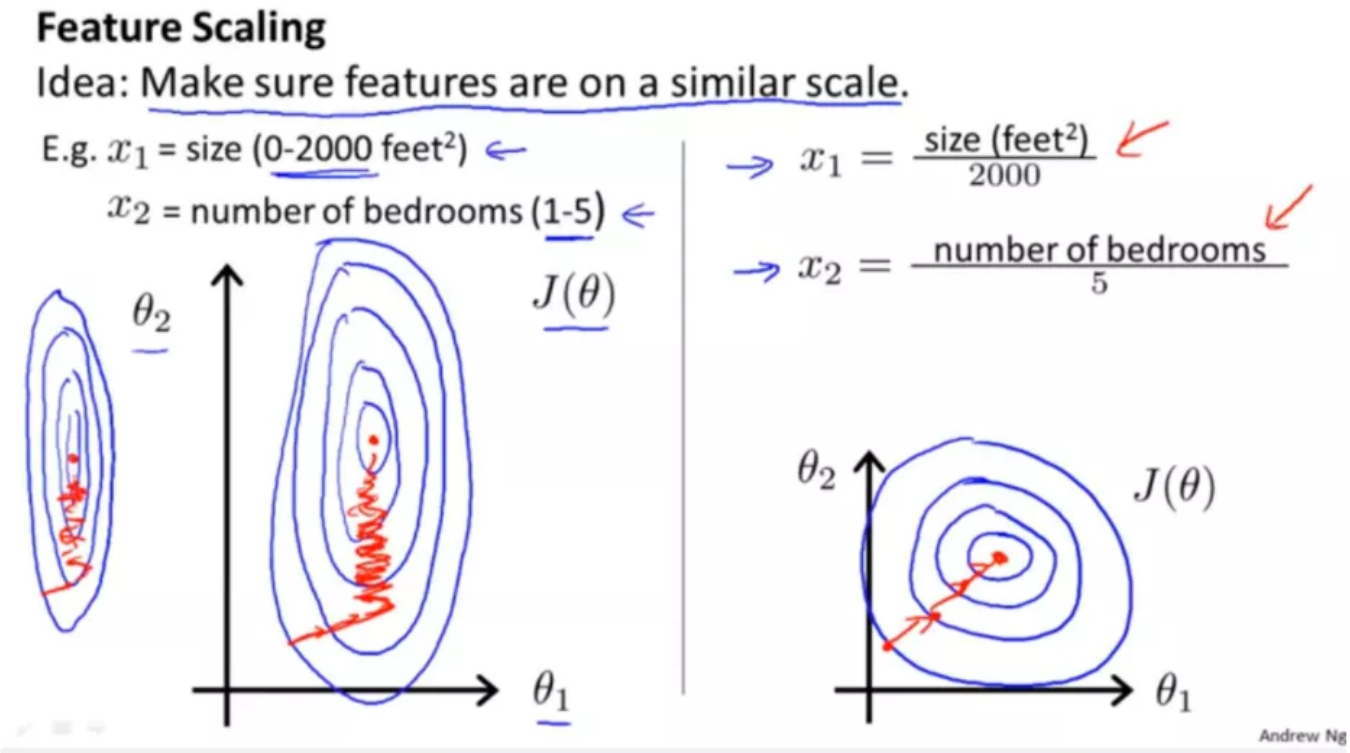
本文解讀了一項數據預處理中的重要技術——特徵歸一化，提出並解答了5個相關問題，同時分析了相關方法和適用場景。

寫在前面

Feature scaling 談到feature scaling的必要性，最常用的2個例子可能是：

- **特徵間的單位（尺度）可能不同**

- 原始特徵下，通過對特徵進行zero-mean and unit-variance變換後，其損失函數的等高線圖更接近圓形，梯度下降的方向震盪更小，收斂更快，如下圖所示，圖片來自Andrew Ng。



Feature Scaling from Andrew Ng

對於feature scaling中最常使用的Standardization，似乎“無腦上”就行了，本文想多探究一些為什麼，

- 常用的feature scaling方法都有哪些?
- 什麼情況下該使用什麼feature scaling方法？有沒有一些指導思想?
- 所有的機器學習算法都需要feature scaling嗎？有沒有例外?

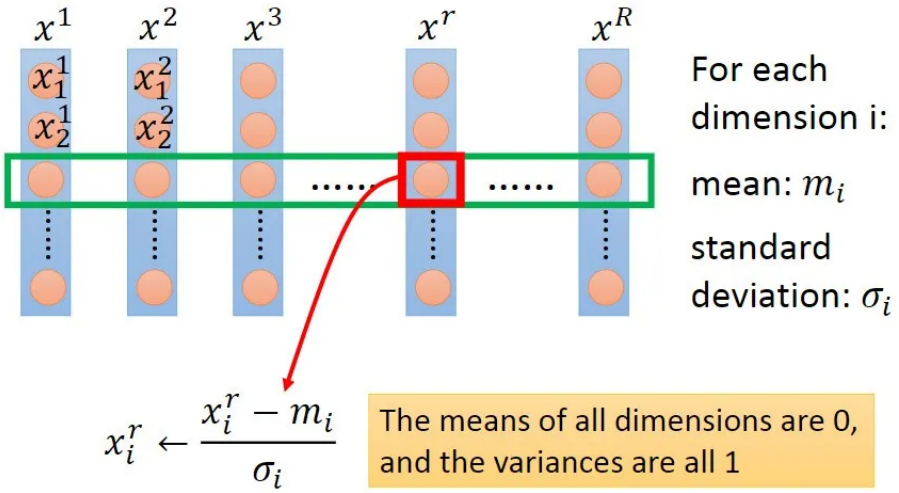
- 損失函數的等高線圖都是橢圓或同心圓嗎？能用橢圓和圓來簡單解釋feature scaling的作用嗎？
- 如果損失函數的等高線圖很複雜，feature scaling還有其他直觀解釋嗎？

根據查閱到的資料，本文將嘗試回答上面的問題。但筆者能力有限，空有困惑，能講到哪算哪吧（微笑）。

常用feature scaling方法

在問為什麼前，先看是什麼。

給定數據集，令特徵向量為x，維數為D，樣本數量為R，可構成D×R的矩陣，一列為一個樣本，一行為一維特徵，如下圖所示，圖片來自Hung-yi Lee pdf-Gradient Descent：



feature matrix

feature scaling的方法可以分成2類，逐行進行和逐列進行。逐行是對每一維特徵操作，逐列是對每個樣本操作，上圖為逐行操作中特徵標準化的示例。

具體地，常用feature scaling方法如下，來自wiki，

- **Rescaling (min-max normalization、range scaling)**

$$x' = a + \frac{(x - \min(x))(b - a)}{\max(x) - \min(x)}$$

將每一維特徵線性映射到目標範圍[a,b]，即將最小值映射為a，最大值映射為b，常用目標範圍為[0,1]和[−1,1]，特別地，映射到[0,1]計算方式為：

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- **Mean normalization**

$$x' = \frac{x - \bar{x}}{\max(x) - \min(x)}$$

將

- **Standardization (Z-score Normalization)**

$$x' = \frac{x - \bar{x}}{\sigma}$$

每維特徵

- **Scaling to unit length**

$$x' = \frac{x}{\|x\|}$$

將每個樣本的特徵向量除以其長度，即對樣本特徵向量的長度進行歸一化，長度的度量常使用的是L2 norm（歐氏距離），有時也會採用L1 norm，不同度量方式的一種對比可以參見論文“CVPR2005-Histograms of Oriented Gradients for Human Detection”。

上述4種feature scaling方式，前3種為逐行操作，最後1種為逐列操作。

容易讓人困惑的一點是指代混淆，Standardization指代比較清晰，但是單說Normalization有時會指代min-max normalization，有時會指代Standardization，有時會指代Scaling to unit length

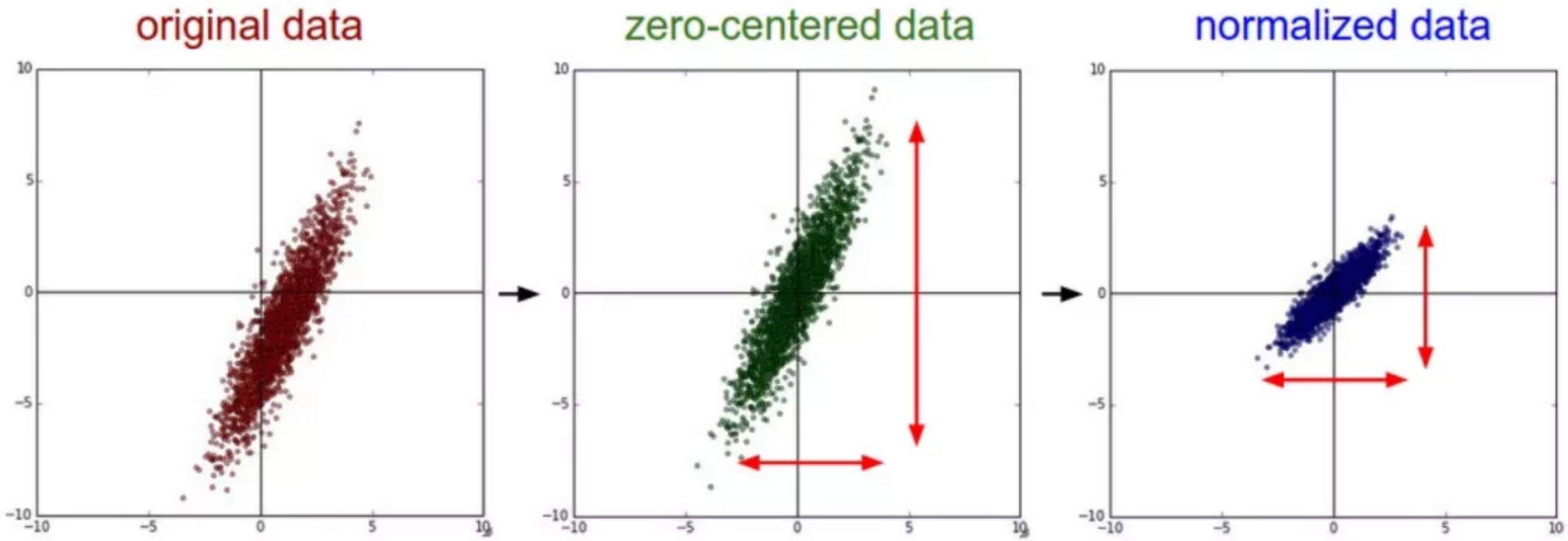
計算方式上對比分析

前3種feature scaling的計算方式為

- **減一個統計量 如果特徵間偏置不同對後續過程有負面影響，則該操作是有益的，可以看成是某種**
- **除以一個統計量 縮放可以使用最大值最小值間的跨度，也可以使用標準差（到中心點的平均距離），前者對outliers敏感，outliers對後者影響與outliers數量和數據集大小有關，outliers越少數據集越大影響越小。**
- **除以長度**

稀疏數據、outliers相關的更多數據預處理內容可以參見scikit learn-5.3. Preprocessing data。

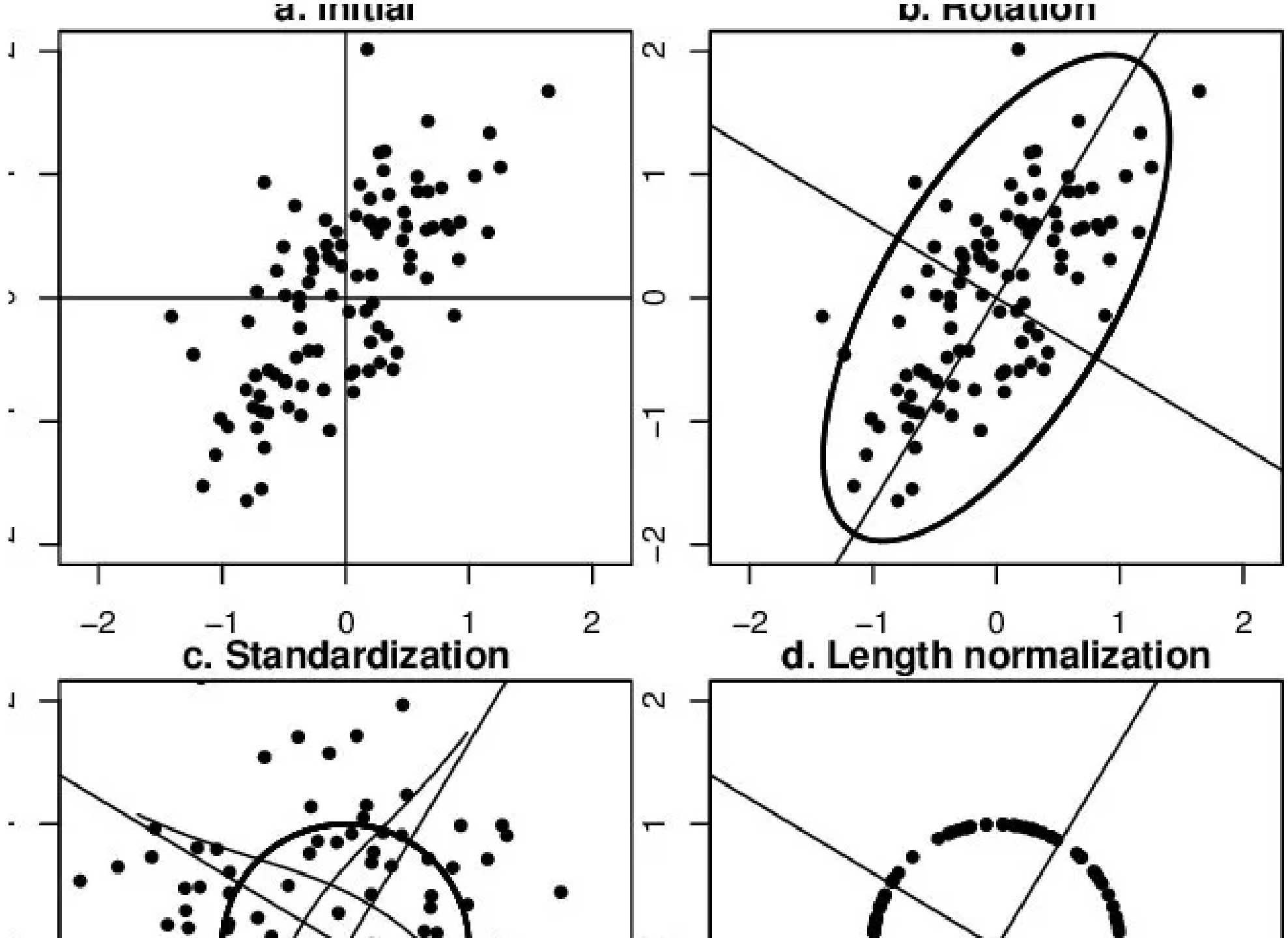
從幾何上觀察上述方法的作用，圖片來自CS231n-Neural Networks Part 2: Setting up the Data and the Loss，zero-mean將數據集平移到原點，unit-variance使每維特徵上的跨度相當，圖中可以明顯看出兩維特徵間存在線性相關性，Standardization操作並沒有消除這種相關性。

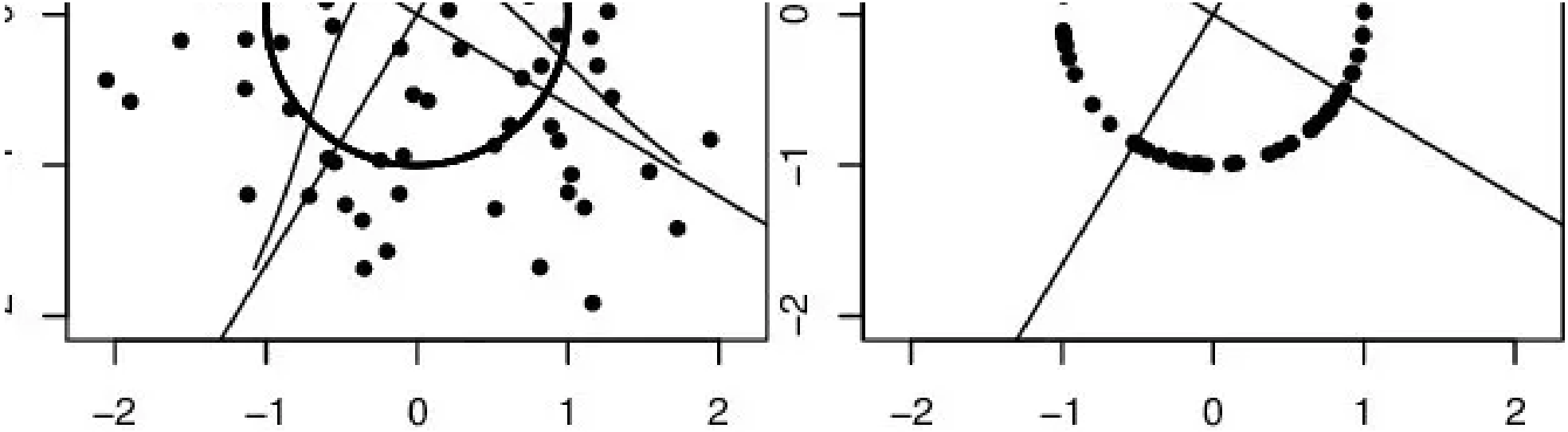


Common data preprocessing pipeline. **Left**: Original toy, 2-dimensional input data. **Middle**: The data is zero-centered by subtracting the mean in each dimension. The data cloud is now centered around the origin. **Right**: Each dimension is additionally scaled by its standard deviation. The red lines indicate the extent of the data - they are of unequal length in the middle, but of equal length on the right.

Standardization

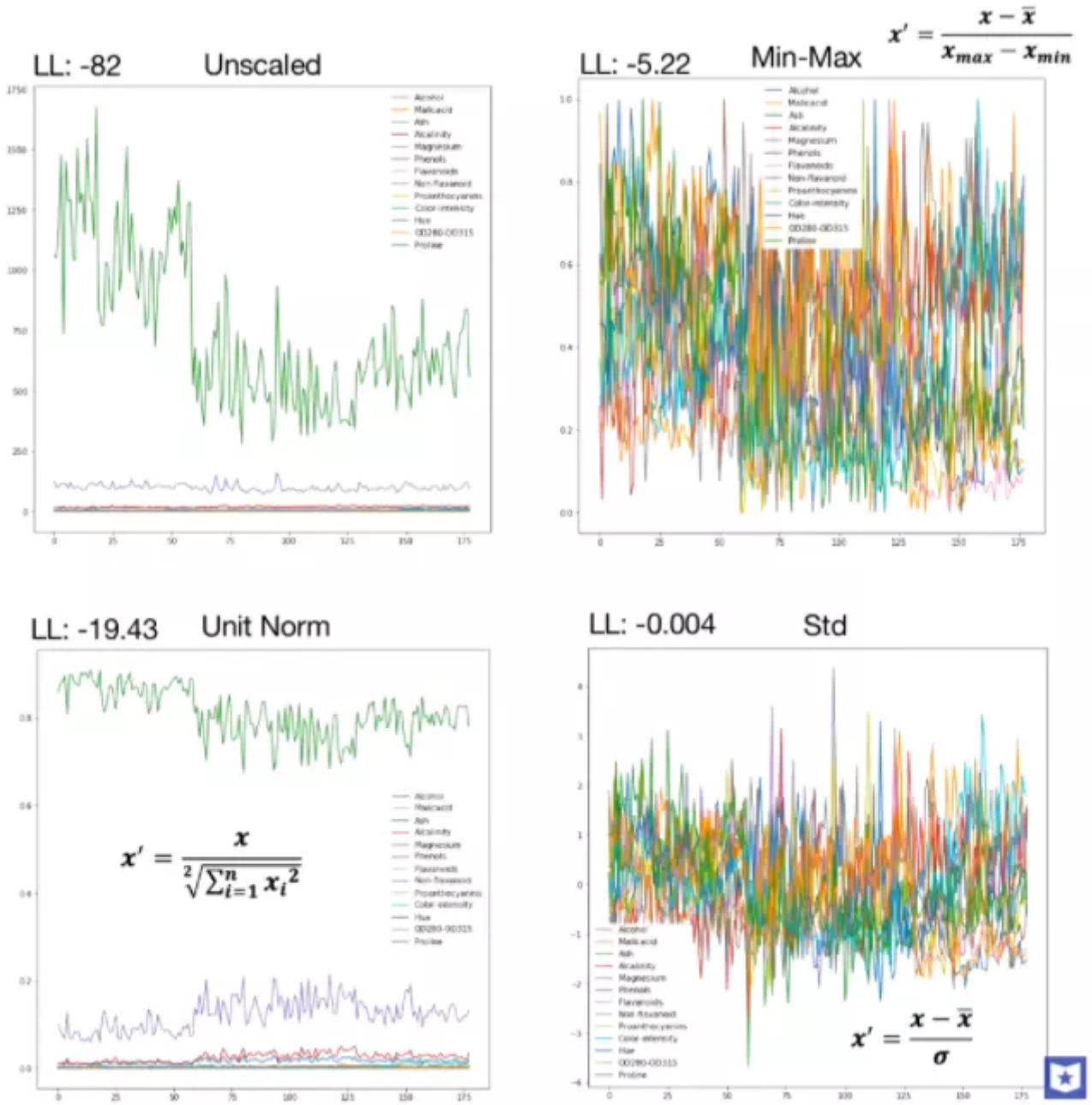
可通過PCA方法移除線性相關性（decorrelation），即引入旋轉，找到新的坐標軸方向，在新坐標軸方向上用“標準差”進行縮放，如下圖所示，圖片來自鏈接，圖中同時描述了unit length的作用——將所有樣本映射到單位球上。


a. Initial
b. Rotation
c. Standardization
d. Length normalization
−2
−1
0
1
2



Effect of the operations of standardization and length normalization

當特徵維數更多時，對比如下，圖片來自youtube，



feature scaling comparison

總的來說，所以，“何時選擇何種方法”取決於待解決的問題，即problem-dependent。

feature scaling 需要還是不需要

下圖來自data school-Comparing supervised learning algorithms，對比了幾個監督學習算法，最右側兩列為是否需要feature scaling。

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	**Algorithm**	**Problem Type**	**Results interpretable by you?**	**Easy to explain algorithm to others?**	**Average predictive accuracy**	**Training speed**	**Prediction speed**	**Amount of parameter tuning needed (excluding feature selection)**	**Performs well with small number of observations?**	**Handles lots of irrelevant features well (separates signal from noise)?**	**Automatically learns feature interactions?**	**Gives calibrated probabilities of class membership?**	**Parametric?**	**Features might need scaling?**	**Algorithm**
2	KNN	Either	Yes	Yes	Lower	Fast	Depends on n	Minimal	No	No	No	Yes	No	Yes	KNN
3	Linear regression	Regression	Yes	Yes	Lower	Fast	Fast	None (excluding regularization)	Yes	No	No	N/A	Yes	No (unless regularized)	Linear regression
4	Logistic regression	Classification	Somewhat	Somewhat	Lower	Fast	Fast	None (excluding regularization)	Yes	No	No	Yes	Yes	No (unless regularized)	Logistic regression
5	Naive Bayes	Classification	Somewhat	Somewhat	Lower	Fast (excluding feature extraction)	Fast	Some for feature extraction	Yes	Yes	No	No	Yes	No	Naive Bayes
6	Decision trees	Either	Somewhat	Somewhat	Lower	Fast	Fast	Some	No	No	Yes	Possibly	No	No	Decision trees
7	Random Forests	Either	A little	No	Higher	Slow	Moderate	Some	No	Yes (unless noise ratio is very high)	Yes	Possibly	No	No	Random Forests
8	AdaBoost	Either	A little	No	Higher	Slow	Fast	Some	No	Yes	Yes	Possibly	No	No	AdaBoost
9	Neural networks	Either	No	No	Higher	Slow	Fast	Lots	No	Yes	Yes	Possibly	No	Yes	Neural networks
10															

Comparing supervised learning algorithms

下面具體分析一下。

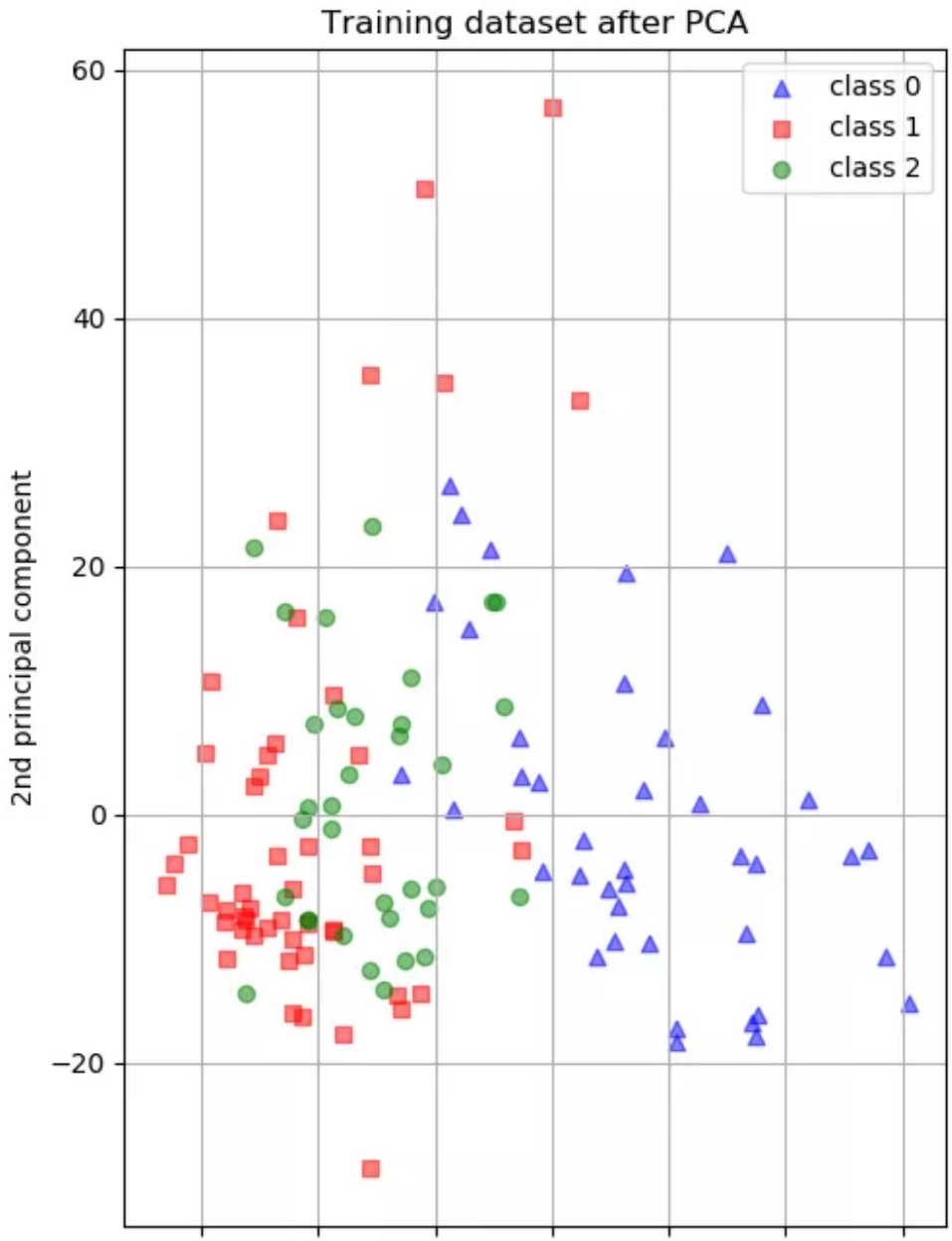
什麼時候需要feature scaling?

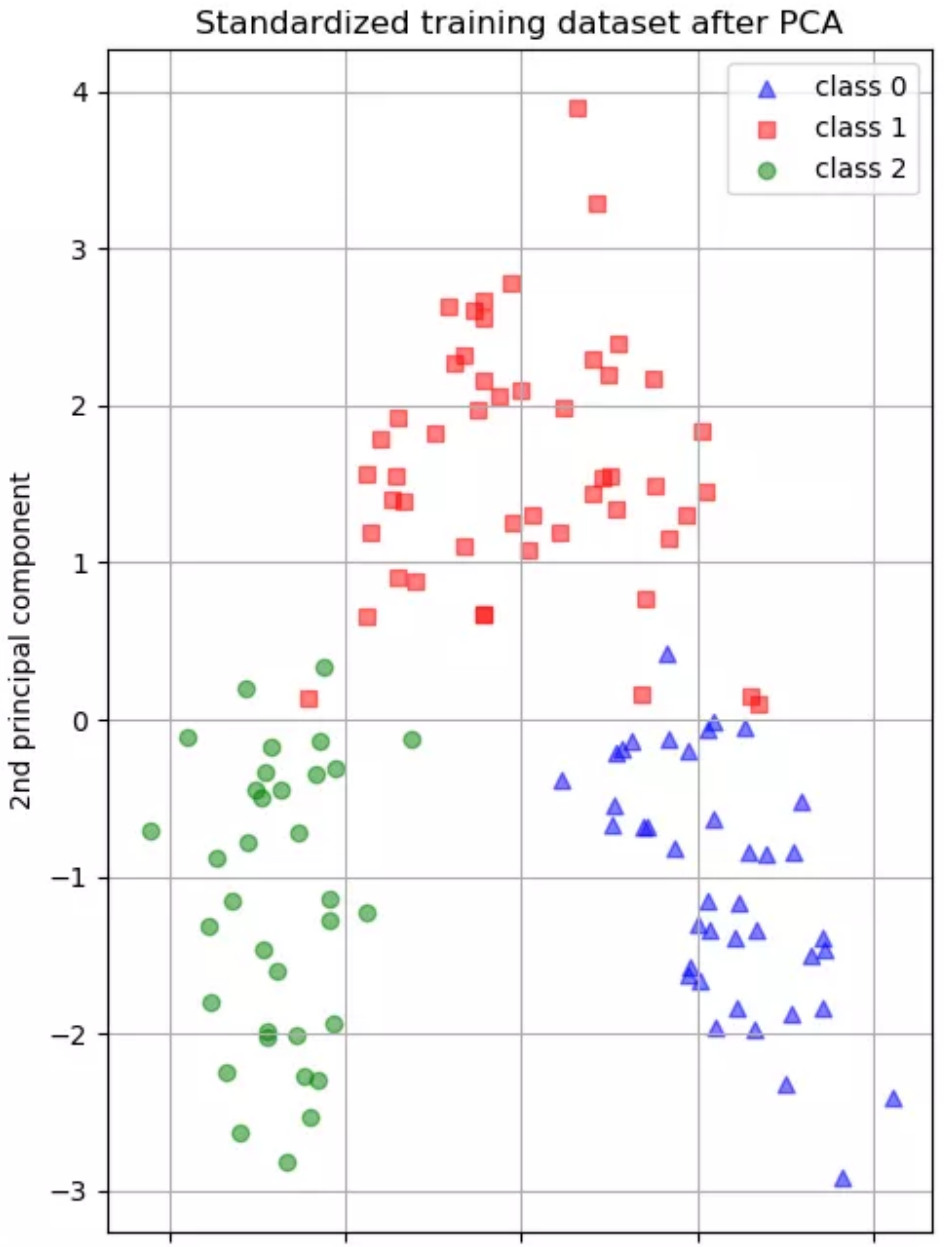
- 涉及或隱含

zero-mean一般可以增加樣本間餘弦距離或者內積結果的差異 在模版匹配中，zero-mean可以明顯提高響應結果的區分度。

就歐式距離而言，

增大尺度的同時也增大了該特徵維度上的方差，PCA算法傾向於關注方差較大的特徵所在的坐標軸方向，其他特徵可能會被忽視，因此，在PCA前做Standardization效果可能更好，如下圖所示，圖片來自scikit learn-Importance of Feature Scaling，





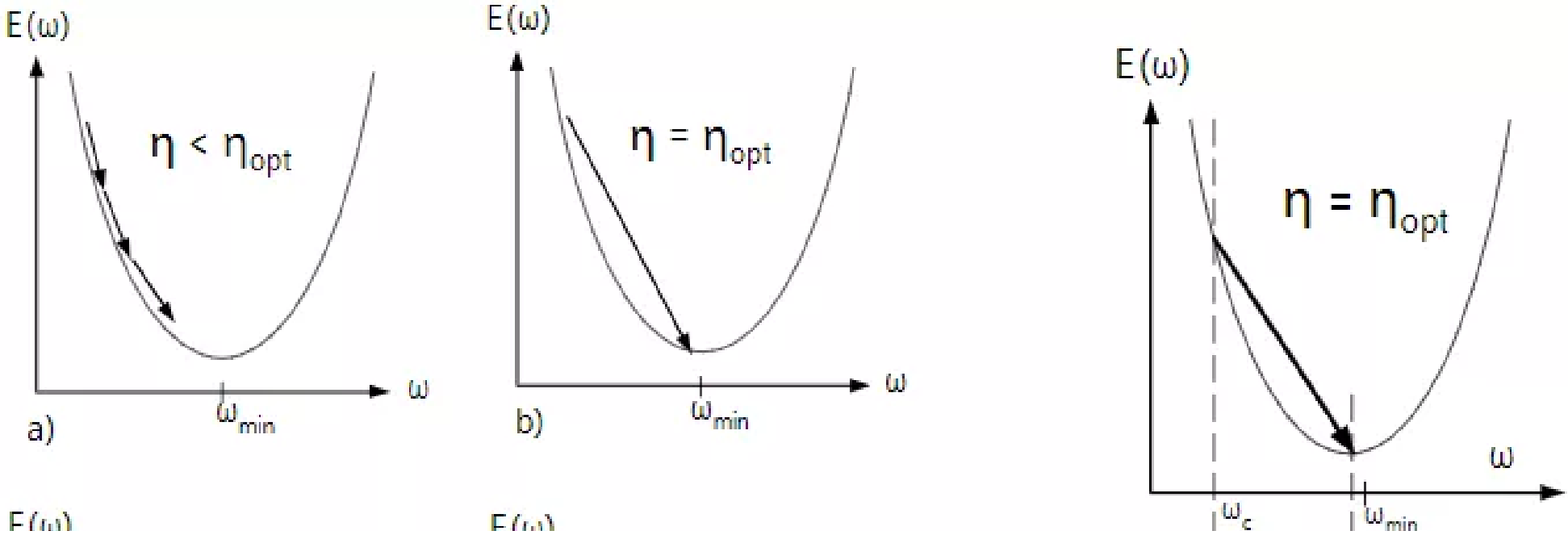
PCA and Standardization

- 損失函數中含有 但是，如果損失函數中含有正則項，如λ||w||^2，λ為超參數，其對w的每一個參數施加同樣的懲罰，但對於某一維特徵xi而言，其scale越大，係數wi越小，其在正則項中的比重就會變小，相當於對wi懲罰變小，即損失函數會相對忽視那些scale增大的特徵，這並不合理，所以需要feature scaling，使損失函數平等看待每一維特徵。

- **梯度下降算法，需要feature scaling 梯度下降的參數更新公式如下，**

$$W(t+1) = W(t) - \eta \frac{dE(W)}{dW}$$

E(W)為損失函數，一維情況下，



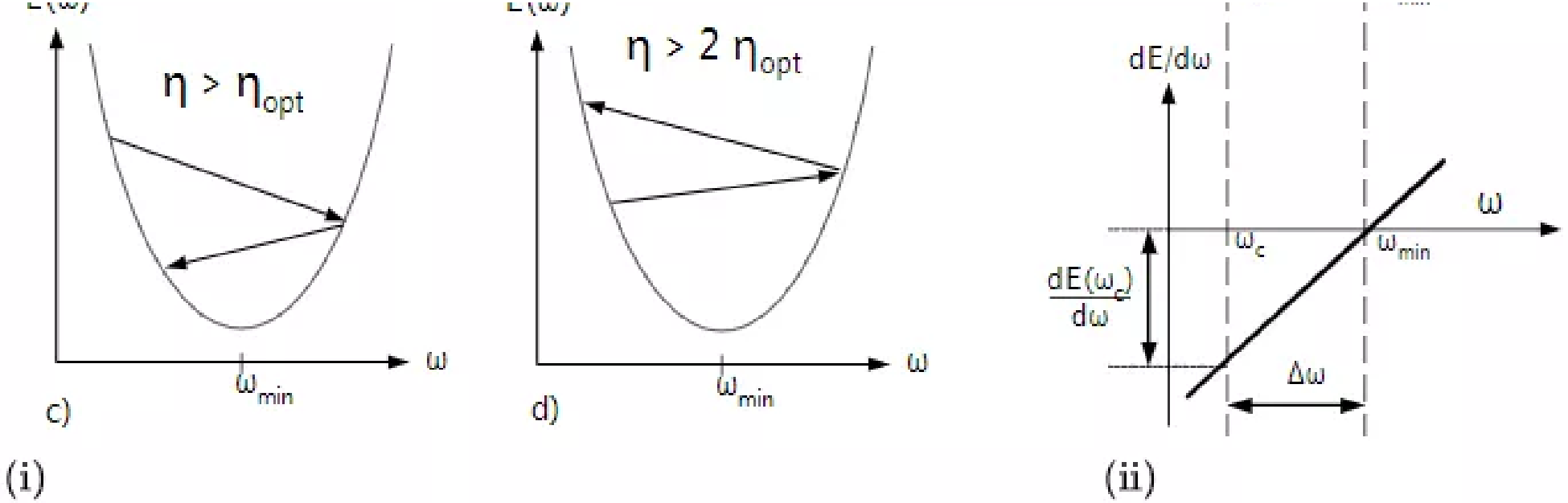


Fig. 6. Gradient descent for different learning rates.

Gradient descent for different learning rates

多維情況下可以分解成多個上圖，每個維度上分別下降，參數W為向量，但學習率只有1個，即所有參數維度共用同一個學習率（暫不考慮為每個維度都分配單獨學習率的算法）。收斂意味著在每個參數維度上都取得極小值，每個參數維度上的偏導數都為0，但是每個參數維度上的下降速度是不同的，為了每個維度上都能收斂，學習率應取所有維度在當前位置合適步長中最小的那個。下面討論feature scaling對gradient descent的作用，

- **zero center與參數初始化相配合，縮短初始參數位置與local minimum間的距離，加快收斂 模型的最終參數是未知的，所以一般隨機初始化，比如從0均值的均勻分佈或高斯分佈中採樣得到，對線性模型而言，其分界面初始位置大致在原點附近，bias經常初始化為0，則分界面直接通過原點。同時，為了收斂，學習率不會很大。而每個數據集的特徵分佈是不一樣的，如果其分佈集中且距離原點較遠，比如位於第一象限遙遠的右上角，分界面可能需要花費很多步驟才能"爬到"數據集所在的位置。所以，無論什麼數據集，先平移到原點，再配合參數初始化，可以保證分界面一定會穿過數據集。此外，**

- 對於採用均方誤差損失LMS的線性模型，損失函數恰為二階，如下圖所示

$$E(W) = \frac{1}{2P}\sum_{p=1}^{P}\left|d^p - \sum_i w_i x_i^p\right|^2$$

不同方向上的下降速度變化不同（二階導不同，曲率不同）將每個維度上的下降分解來看，給定一個下降步長，如果不夠小，有的維度下降的多，有的下降的少，有的還可能在上升，損失函數的整體表現可能是上升也可能是下降，就會不穩定。scaling後不同方向上的曲率相對更接近，更容易選擇到合適的學習率，使下降過程相對更穩定。

- 另有從Hessian矩陣特徵值以及condition number角度的理解，詳見Lecun paper-Efficient BackProp中的Convergence of Gradient Descent一節，有清晰的數學描述，同時還介紹了白化的作用——解除特徵間的線性相關性，使每個維度上的梯度下降可獨立看待。

- 文章開篇的橢圓形和圓形等高線圖，僅在採用均方誤差的線性模型上適用，其他損失函數或更複雜的模型，如深度神經網絡，

- 對於損失函數不是均方誤差的情況，只要權重w與輸入特徵x間是相乘關係，損失函數對w的偏導必然含有因子x，w的梯度下降速度就會受到特徵x尺度的影響。理論上為每個參數都設置上自適應的學習率，可以吸收掉x尺度的影響，但在實踐中出於計算量的考慮，往往還是所有參數共用一個學習率，此時x尺度不同可能會導致不同方向上的下降速度懸殊較大，學習率不容易選擇，下降過程也可能不穩定，通過scaling可對不同方向上的下降速度有所控制，使下降過程相對更穩定。

- 對於傳統的神經網絡，對輸入做feature scaling也很重要，因為採用sigmoid等有飽和區的激活函數，如果輸入分佈範圍很廣，參數初始化時沒有適配好，很容易直接陷入飽和區，導致 但自從有了Batch Normalization，每次線性變換改變特徵分佈後，都會重新進行Normalization，似乎可以不太需要對網絡的輸入進行feature scaling了？但習慣上還是會做feature scaling。

什麼時候不需要Feature Scaling?

與距離計算無關的概率模型，不需要feature scaling，比如Naive Bayes；

與距離計算無關的基於樹的模型，不需要feature scaling，比如決策樹、隨機森林等，樹中節點的選擇只關注當前特徵在哪里切分對分類更好，即只在意特徵內部的相對大小，而與特徵間的相對大小無關。

小結

這篇文章寫得十分艱難，一開始以為蠻簡單直接，但隨著探索的深入，冒出的問號越來越多，打破了很多原來的“理所當然”，所以，在寫的過程中不停地做加法，很多地方想解釋得盡量直觀，又不想照搬太多公式，但自己的理解又不夠深刻，導致現在敘述這麼冗長，希望以後在寫文時能更專注更精煉。

參考文獻

1.wiki-Feature scaling
2.wiki-Backpropagation
3.[Hung-yi Lee pdf-Gradient Descent]
(<http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2016/Lecture/Gradient Descent (v2).pdf>)
4.quora-Why does mean normalization help in gradient descent?
5.scikit learn-Importance of Feature Scaling
6.scikit learn-5.3. Preprocessing data

推薦閱讀 （點擊標題可跳轉閱讀）

神經網絡入門

統計學無用了?

我的深度學習之路

23個優秀的機器學習數據集

6行代碼！用Python將PDF轉為word

台大美女教授陳縕儂：《應用深度學習》

李宏毅《機器學習》視頻教程PPT

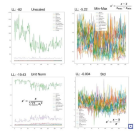
老鐵，三連支持一下，好嗎？↓↓↓

閱讀原文

喜歡此內容的人還喜歡

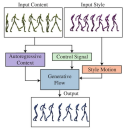
深入探討：為什麼要做特徵歸一化/標準化?

圖靈人工智能



計圖開源：基於標準化生成流的人體運動風格遷移方法

圖形學與幾何計算



網絡架構之爭：三大主流架構對決，誰是王者？深入思考CNN、Transformer與MLP

機器學習實驗室