

為什麼要做特徵的歸一化/標準化？

OpenCV與AI深度學習 2022-09-24 09:17 發表於重慶

收錄於合集

#特徵提取 3 #歸一化 1

點擊下方**卡片**，關注“**OpenCV與AI深度學習**”

視覺/圖像重磅乾貨，第一時間送達！



OpenCV與AI深度學習

專注計算機視覺、深度學習和人工智能領域乾貨、應用、行業資訊的分享交流！

168篇原創內容

公眾號

作者 | shine-lee

鏈接 | <https://blog.csdn.net/blogshinelee/article/details/102875044>

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**，比如身高和體重，比如攝氏度和華氏度，比如房屋面積和房間數，一個特徵的變化範圍可能是 $[1000, 10000]$ ，另一個特徵的變化範圍可能是 $[-0.1, 0.2]$ ，在進行距離有關的計算時，單位的不同會導致計算結果的不同，尺度大的特徵會起決定性作用，而尺度小的特徵其作用可能會被忽略，**為了消除特徵間單位和尺度差異的影響，以對每維特徵同等看待，需要對特徵進行歸一化**。
- 原始特徵下，**因尺度差異，其損失函數的等高線圖可能是橢圓形**，梯度方向垂直於等高線，下降會走zigzag路線，而不是指向local minimum。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

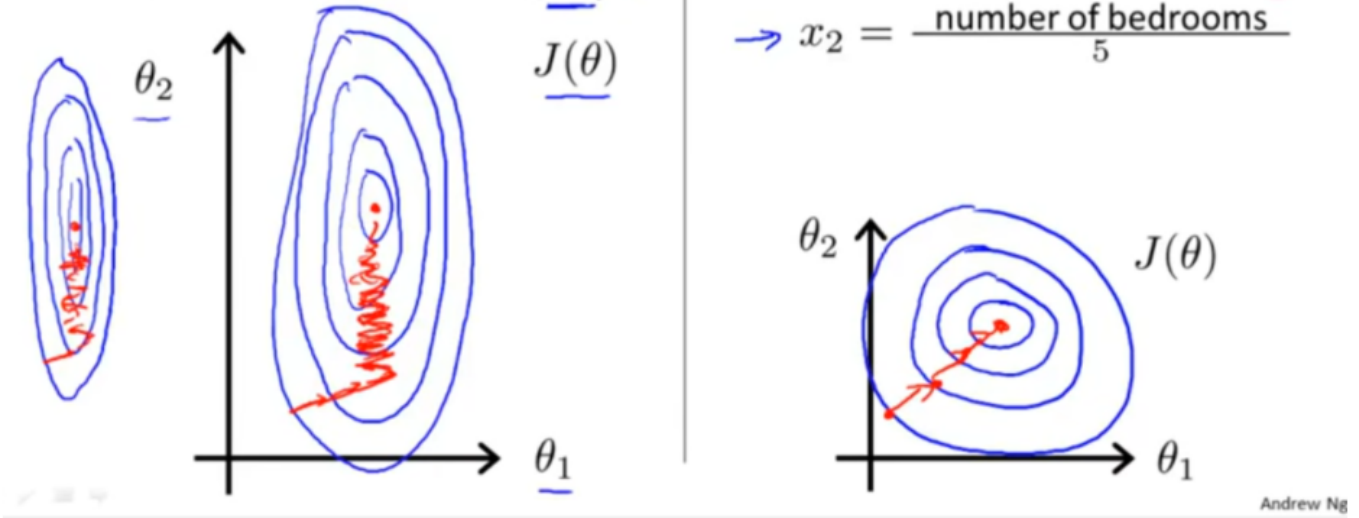
Idea: Make sure features are on a similar scale.

E.g. $x_1 = \text{size (0-2000 feet}^2\text{)}$ ←

$x_2 = \text{number of bedrooms (1-5)}$ ←

→ $x_1 = \frac{\text{size (feet}^2\text{)}}{2000}$ ←

→ $x_2 = \frac{\text{number of bedrooms}}{5}$ ←



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，

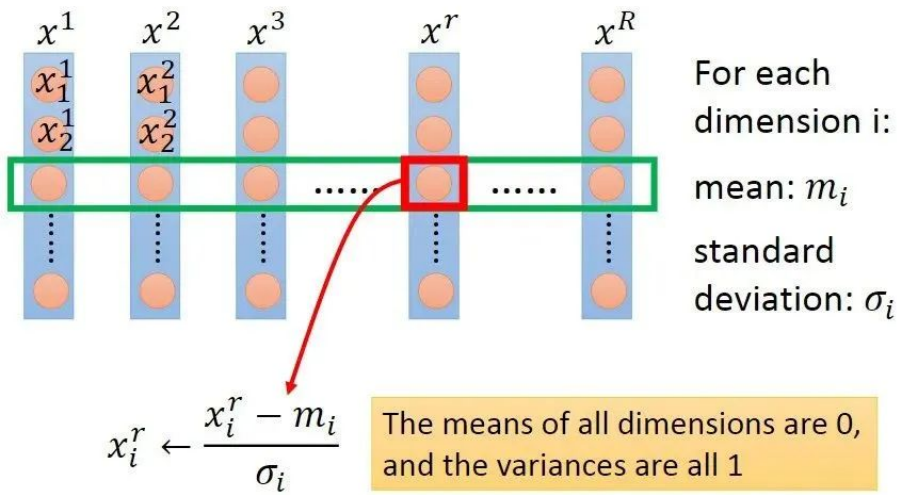
- 常用的feature scaling方法都有哪些？
- 什麼情況下該使用什麼feature scaling方法？有沒有一些指導思想？
- 所有的機器學習算法都需要feature scaling嗎？有沒有例外？
- 損失函數的等高線圖都是橢圓或同心圓嗎？能用橢圓和圓來簡單解釋feature scaling的作用嗎？
- 如果損失函數的等高線圖很複雜，feature scaling還有其他直觀解釋嗎？

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 $D \times R$ 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**均值映射為0**，同時用最大值最小值的差對特徵進行歸一化，一種更常見的做法是用標準差進行歸一化，如下。

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

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

每維特徵**0均值1方差 (zero-mean and unit-variance)**。

- **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

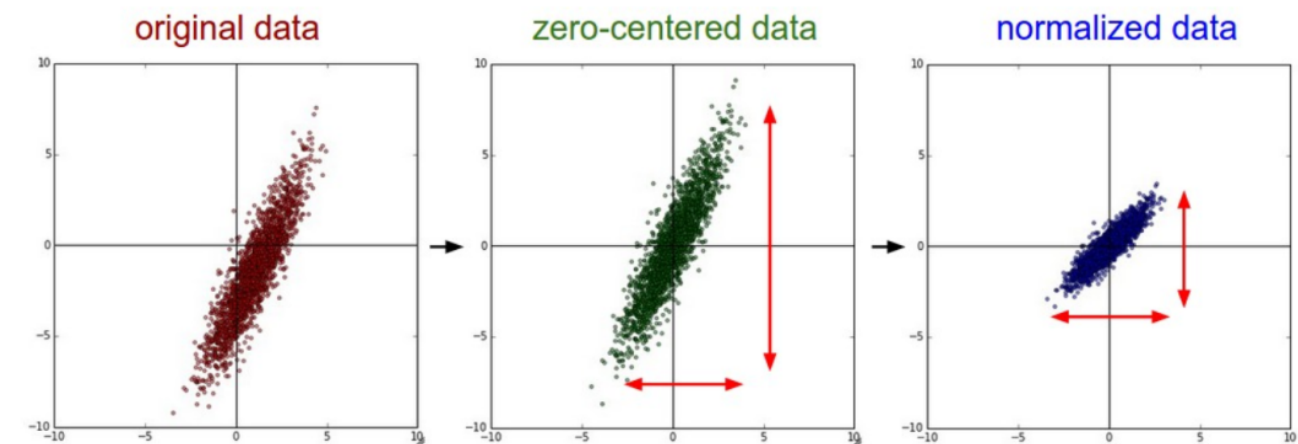
前3種feature scaling的計算方式為**減一個統計量再除以一個統計量**，最後1種為**除以向量自身的長度**。

- **減一個統計量**可以看成選哪個值作為原點，是最小值還是均值，並將整個數據集平移到這個新的原點位置。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**偏置無關操作**；如果原始特徵值有特殊意義，比如稀疏性，該操作可能會破壞其稀疏性。
- **除以一個統計量**可以看成在坐標軸方向上對特徵進行縮放，用於降低特徵尺度的影響，可以看成是某種**尺度無關操作**。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。

- 除以長度相當於把長度歸一化，把所有樣本映射到單位球上，可以看成是某種長度無關操作，比如，詞頻特徵要移除文章長度的影響，圖像處理中某些特徵要移除光照強度的影響，以及方便計算餘弦距離或內積相似度等。

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。

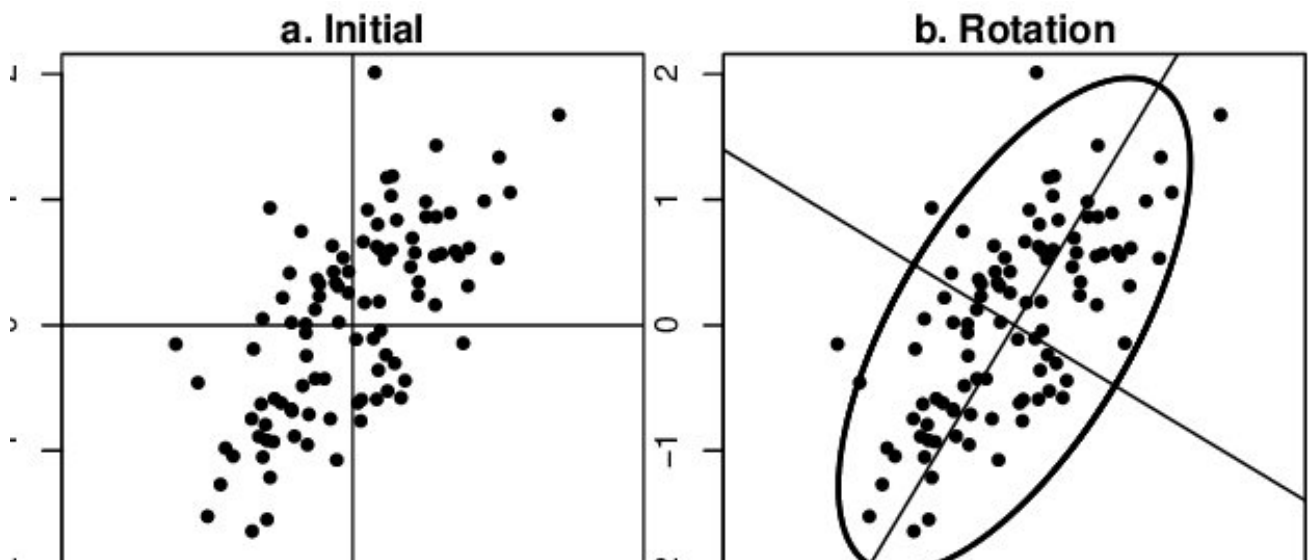
從幾何上觀察上述方法的作用，圖片來自CS231n-Neural Networks Part 2: Setting up the Data and the Loss，zero-mean將數據集平移到原點，unit-variance使每維特徵上的跨度相當，圖中可以明顯看出兩維特徵間存在線性相關性，Standardization操作並沒有消除這種相關性。

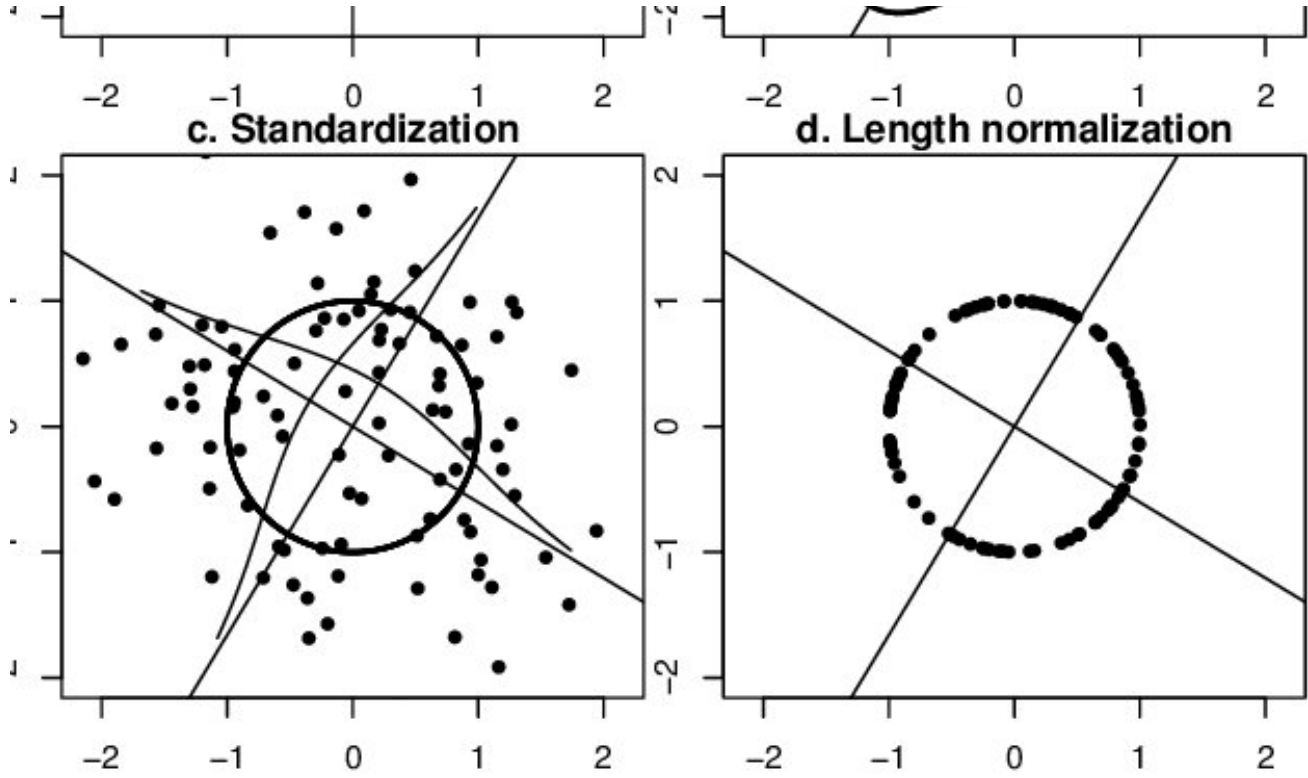


Common data preprocessing pipeline. Left: Original toy, 2-dimensional input data. Middle: The data is zero-centered by subtracting the mean in each dimension. The data cloud is now centered around the origin. Right: Each dimension is additionally scaled by its standard deviation. The red lines indicate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

Standardization

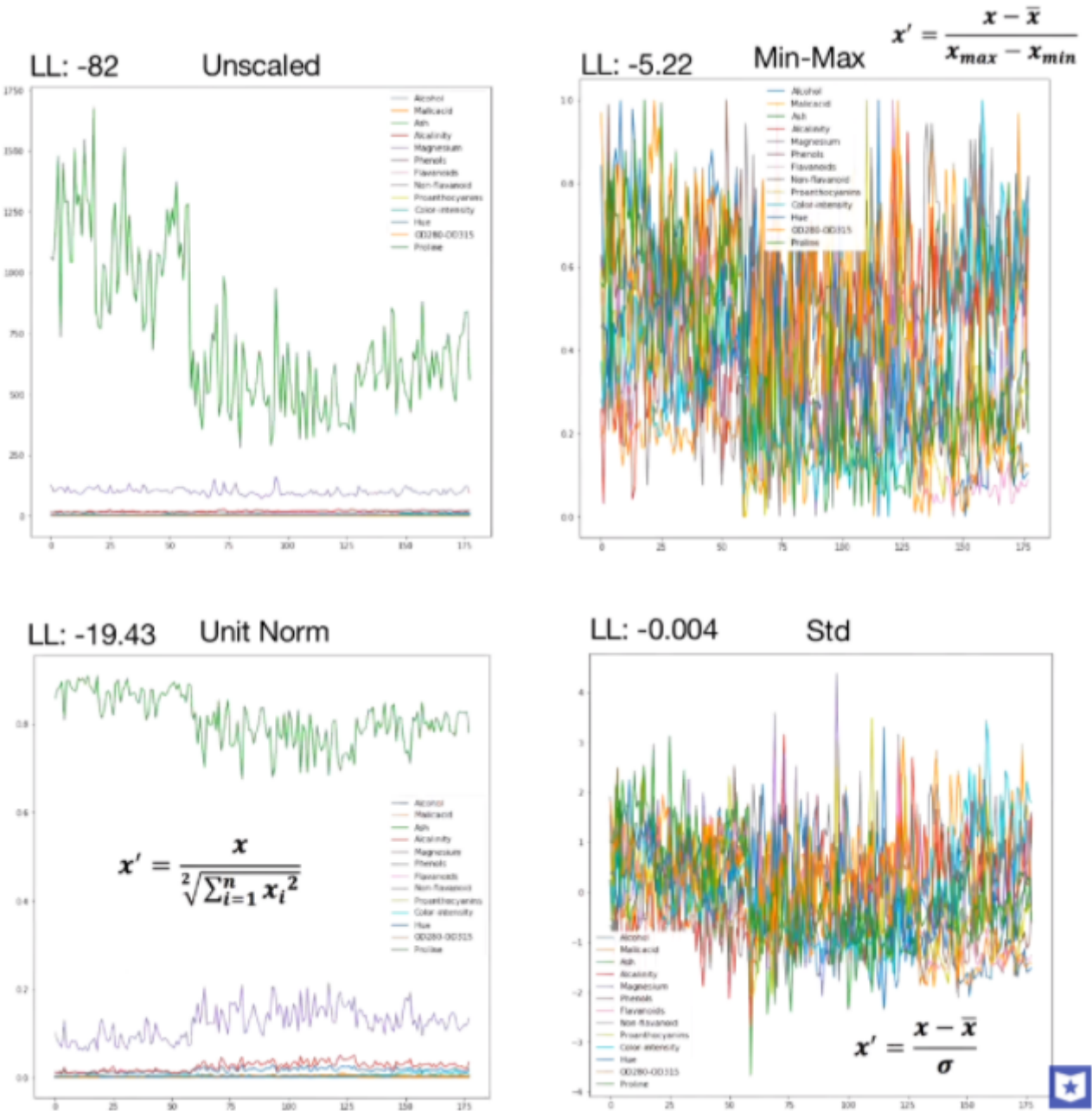
可通過PCA方法移除線性相關性 (decorrelation)，即引入旋轉，找到新的坐標軸方向，在新坐標軸方向上用“標準差”進行縮放，如下圖所示，圖片來自鏈接，圖中同時描述了unit length的作用——將所有樣本映射到單位球上。





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，歸一化/標準化的目的是為了獲得某種“無關性”——偏置無關、尺度無關、長度無關.....當歸一化/標準化方法背後的物理意義和幾何含義與當前問題的需要相契合時，其對解決該問題就有正向作用，反之，就會起反作用。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	Linear regression	Classification	Somewhat	Somewhat	Lower	Fast	Fast	None (excluding regularization)	Yes	No	No	Yes	Yes	No (unless regularized)	Linear regression

	Logistic regression	Naive Bayes	Linear models	Linear models	Support	Fast (excluding feature extraction)	Fast	regression only	Yes	Yes	Yes	Yes	Yes	regression only	Logistic regression
5	Naive Bayes	Classification	Somewhat	Somewhat	Lower	Fast (excluding feature extraction)	Fast	Some for feature extraction	Yes	No	No	No	Yes	No	Naive Bayes
6	Decision trees	Either	Somewhat	Somewhat	Lower	Fast	Fast	Some	No	Yes	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。

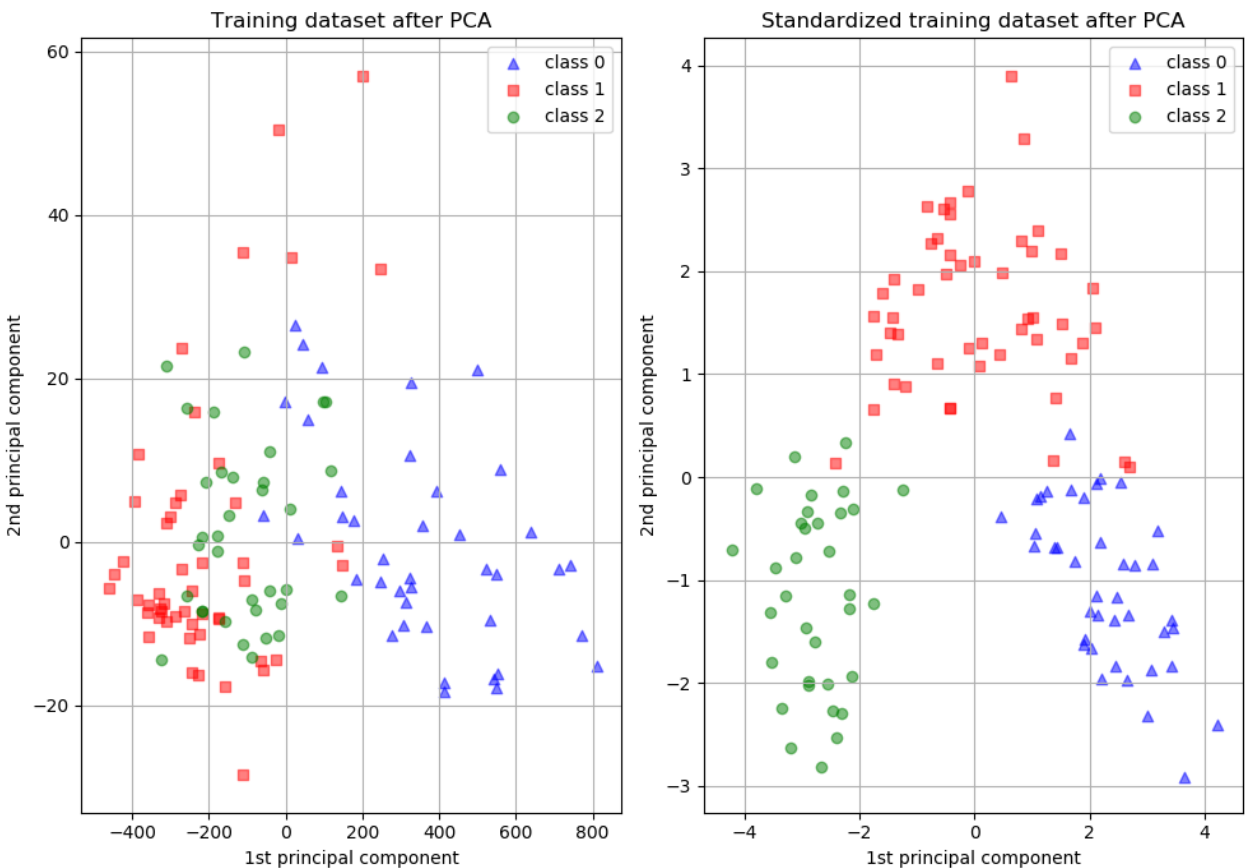
什麼時候需要feature scaling？

- 涉及或隱含**距離計算**的算法，比如K-means、KNN、PCA、SVM等，一般需要feature scaling，因為：

zero-mean一般可以增加樣本間餘弦距離或者內積結果的差異，區分力更強，假設數據集集中分佈在第一象限遙遠的右上角，將其平移到原點處，可以想像樣本間餘弦距離的差異被放大了。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，增大某個特徵的尺度，相當於增加了其在距離計算中的權重，如果有明確的先驗知識表明某個特徵很重要，那麼適當增加其權重可能有正向效果，但如果沒有這樣的先驗，或者目的就是想知道哪些特徵更重要，那麼就需要先feature scaling，對各維特徵等而視之。

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**正則項**時，一般需要feature scaling：對於線性模型 $y=wx+b$ 而言， x 的任何線性變換（平移、放縮），都可以被 w 和 b “吸收”掉，理論上，不會影響模型的擬合能力。但是，如果損失函數中含有正則項，如 $\lambda ||w||^2$ ， 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 x_i 而言，其scale越大，係數 w_i 越小，其在正則項中的比重就會變小，相當於對 w_i 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，收斂速度取決於：參數的初始位置到local minima的距離，以及學習率 η 的大小。一維情況下，在local minima附近，不同學習率對梯度下降的影響如下圖所示：

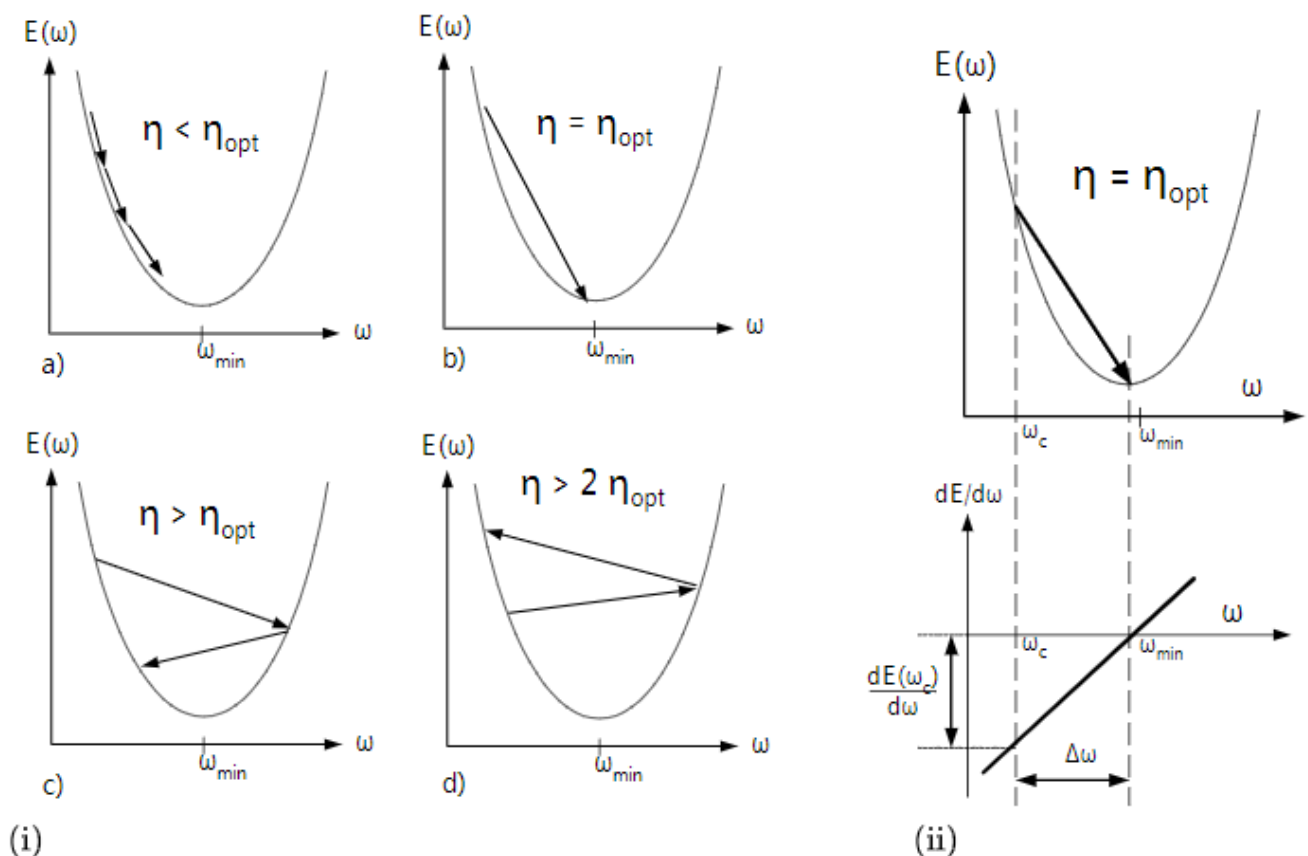


Fig. 6. Gradient descent for different learning rates.

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，**outliers**常分佈在數據集的外圍，與分界面從外部向內挪動相比，從中心區域開始挪動可能受outliers的影響更小。

- 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），恰由輸入的協方差矩陣決定，通過scaling改變了損失函數的形狀，減小不同方向上的曲率差異。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，**損失函數的error surface可能很複雜，並不能簡單地用橢圓和圓來刻畫**，所以用它來解釋feature scaling對所有損失函數的梯度下降的作用，似乎過於簡化，見Hinton video-3.2 The error surface for a linear neuron。

- 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**梯度消失**，所以，需要對輸入做Standardization或映射到 $[0, 1]$ 、 $[-1, 1]$ ，配合精心設計的參數初始化方法，對值域進行控制。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](http://speech.ee.ntu.edu.tw/~tlkagk/courses/ML_2016/Lecture/Gradient%20Descent%20(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



本文僅做學術分享，如有侵權，請聯繫刪文。

—THE END—

计算机视觉与深度学习·聚焦行业最前沿

- 机器视觉/深度学习/3D视觉/人工智能
- 硬核干货/实战应用/行业信息/前沿速递

商务合作：

投稿咨询：

学习交流：



长按扫描右侧二维码关注"OpenCV与AI深度学习"公众号



OpenCV与AI深度学习

OpenCV&AI&DL



覺得有用，麻煩給個贊和在看



收錄於合集#特徵提取 3

下一篇·特徵提取：傳統算法vs 深度學習

喜歡此內容的人還喜歡

精通機器學習必須掌握的20大算法

小白學視覺



YOLOP v2來啦| YOLOv7結合YOLOP的多任務版本，超越YOLOP以及HybridNets

集智書僮

YOLOP v2來啦!
7 結合 YOLOP 的多任務
YOLOP、HybridNets 進

機器學習生物醫學圖像處理方向研究前景如何？

小白學視覺

