

Cok Degiskenli Normal Numaralari (Multivariate Normal Tricks)

Cok degiskenli normal dagilimlarla is yaparken, mesela Gaussian karisimleri kullanirken, bazi numaralari bilmek faydali olabiliyor. Bunlardan birincisi $(x - \mu)^T \Sigma^{-1} (x - \mu)$ hesabini yapmaktir, diger log-toplam-exp numarasi (logsumexp trick) diye bilinen hesaptir.

Birinciden baslayalim, daha kisalastirmak icin $y = x - \mu$ diyelim, yani $y^T \Sigma^{-1} y$ olsun. Simdi bu formulde bir ters alma (inversion) isleminin oldugunu goruyoruz. Fakat bu islem oldukca pahali bir islem olarak bilinir, hele hele boyutlarin yuk-seldigi durumlardan (binler, onbinler), kovaryansi temsil eden Σ , $n \times n$ olacaktır. Acaba tersini almayi baska bir sekilde gerceklestiremez miyiz?

Σ matrisi bir kovaryans matrisi oldugu icin simetrik, pozitif yari kesin bir matristir. Bu tur matrislerin Cholesky ayristirmasinin oldugunu biliyoruz ve bu islem cok hizli yapilabiliyor. O zaman

$$\Sigma = LL^T$$

ki L matrisi alt-ucgensel (lower triangular) bir matristir,

$$\Sigma^{-1} = (LL^T)^{-1}$$

$$= L^{-T} L^{-1}$$

Bunu temel alarak iki taraftan y 'leri geri koyalik,

$$y^T \Sigma^{-1} y = y^T L^{-T} L^{-1} y$$

Bilindigi gibi lineer cebirde istedigimiz yere parantez koyabiliriz,

$$= (y^T L^{-T}) L^{-1} y$$

Parantezden bir sey in devrigi gibi temsil edersek, parantez icindekilerin sirasi degisir ve tek tek devrigi alinir,

$$= (L^{-1} y)^T L^{-1} y$$

$$= |L^{-1} y|^2$$

Ustteki ifadede $|\cdot|$ icindeki kisim $Ax = b$ durumundaki x 'in en az kareler cozumu olan $A^{-1}b$ 'ye benzemiyor mu? Evet. Bu durumda her standart sayisal kutuphanede mevcut bir cagri ile $L^{-1}y$ hesabini yapabiliriz, bu cagrlar perde arkasinda ters

alma isleminde kacinarak bir suru optimizasyon yaparak sonuca erismektedirler. Ustune ustluk L durumunda bu cok daha hizli olacaktır, cunku L alt-ucgensel oldugu icin cozum geriye deger gecirmek (backsubstitution) ile aninda bulunabilir.

Demek ki $y^T \Sigma^{-1} y$ hesabi icin once Σ uzerinde Cholesky aliyoruz, sonra $L^{-1} y$ coz-
duruyoruz. Elde edilen degerin noktasal carpimini alinca Σ 'nin tersini elde etmis
olacagiz. Ornek, once uzun yoldan,

```
import numpy.linalg as lin
Sigma = np.array([[10., 2.], [2., 5.]])
y = np.array([[1.], [2.]])
print np.dot(np.dot(y.T, lin.inv(Sigma)), y)

[[ 0.80434783]]
```

Simdi Cholesky ve en az kareler uzerinden

```
L = lin.cholesky(Sigma)
x = lin.lstsq(L, y)[0]
print np.dot(x.T, x)

[[ 0.80434783]]
```

Ayni sonuca eristik.

log-toplam-exp

Bu numaranin ilk kısmi nisbeten basit. Bazi yapay ogrenim algoritmaları icin
olasilik degerlerinin birbiriyle carpilmasi gerekiyor, mesela

$$r = p_1 \cdot p_2 \dots p_n$$

Olasiliklar 1'den kucuk oldugu icin 1'den kucuk degerlerin carpimi asiri ku-
culebilir, ve küçüklüğün tasmasi (underflow) ortaya cikabilir. Eger carpim yerine
log alirsak, carpimlar toplama donusur, sonra sonucu exp ile tersine ceviris, ve
log'u alinan degerler cok kuculmez, carpma yernie toplama islemi kullanildigi
icin de nihai deger de kucukluge dogru tasmaz.

$$\log r = \log p_1 + \log p_2 + \dots + \log p_n$$

$$r = \exp(\log p_1 + \log p_2 + \dots + \log p_n)$$

Bir diger durum icinde exp ifadesi tasiyan bir olasilik degerinin cok kucuk degerler
tasiyabilmesidir. Mesela cok degiskenli Gaussian karisimleri icin alttaki gibi bir
hesap surekli yapilir,

$$= \sum_i w_i \frac{1}{(2\pi)^{k/2} \det(\Sigma)^{1/2}} \exp \left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\}$$

ki $0 \leq w_i \leq 1$ şeklinde bir ağırlık değeridir. Üstteki formülün cöşünlölkle \log 'u alınır, ve, mesela bir örneğ üzerinde görürsek (ve ağırlıkları bir kenara bırakırsak),

$$\log(e^{-1000} + e^{-1001})$$

gibi hesaplar olabilir. Üstteki değerler tamamen uyduruk denemez, uygulamalarda pek çok kez karşımıza çıkan değerler bunlar. Her neyse, eğer üstteki ifadeyi kodla hesaplırsak,

```
print np.log(np.exp(-1000) + np.exp(-1001))  
-inf
```

Bu durumdan kurtulmak için bir numara sudur; \exp ifadeleri arasında en büyük olanını dışarı çekersiniz, ve \log 'lar çarpımı toplam yapar,

$$\log(e^{-1000}(e^0 + e^{-1}))$$

$$-1000 + \log(1 + e^{-1})$$

Bunu hesaplırsak,

```
print -1000 + np.log(1+np.exp(-1))  
-999.686738312
```

Bu numaranın yaptığı nedir? Maksimumu dışarı çekerek en az bir değerin küçüklüğü tasmmasını garantilemiş oluyoruz. Geri kalan terimler içinde yine asiri ufaanlar terimler olabilir, fakat bunlar eğer tasmması diğer değerler varsa onlara eklenecektir, yani eriyip gidecektir. Tüm kalan terimler asiri ufalırsa daha önce dediğimiz gibi, hala elimizde hesap için bir değer geçmiş olacak.

Numerical Recipes, 3rd Edition

<http://makarandtapaswi.wordpress.com/2012/07/18/log-sum-exp-trick/>