Google Nasil Isler?

Ozdeger/Vektor Hesabinda Ust Metot (Power Method)

Diyelim ki bir A matrisinin, ki $A \in \mathbb{R}^{n \times n}$, ozdegerleri $\lambda_1, ..., \lambda_n$ ve ozvektorleri $v_1, .., v_n$ olarak verilmis. Bu demektir ki her $i = 1, .., n$ icin $Av_i = \lambda_i v_i$.

Farzedelim ki bu matrisin tum ozvektorleri bir "ozbaz (eigenbasis)" olusturuyor ve bu baz ile $\mathbb{R}^n$'deki herhangi bir vektoru temsil edebiliyoruz. Yine farzedelim ki $|\lambda_1| > |\lambda_2| > .. > |\lambda_n|$. Biz bu yazida λ_1'e baskin (dominant) ozdeger diyecegiz.

Simdi herhangi bir $v_0 \in \mathbb{R}^n$'i alalim. Usttekiler isiginda $\mu_1, .., \mu_n$ olarak katsayilar olmalidir, ki

$$v_o = \mu_1 v_1 + .. + \mu_n v_n$$

cunku ozvektorler bir baz olusturuyorlar. Simdi her iki tarafi soldan A ile carpalim, ayrica $Av_i = \lambda_i v_i$ esitliginden hareketle ustteki esitligin sag tarafini alip ucuncu bir esitlik olarak en sagda yazalim,

$$Av_o = \mu_1 A v_1 + .. + \mu_n A v_n = \mu_1 \lambda_1 v_1 + ... + \mu_n \lambda_n v_n$$

Simdi ustteki ifadeyi A ile bir daha, hatta birkac defa carpalim, diyelim toplam m kere carpmis olalim,

$$A^m v_o = \mu_1 A^m v_1 + .. + \mu_n A^m v_n = \mu_1 \lambda_1^m v_1 + ... + \mu_n \lambda_n^m v_n \tag{1}$$

En sagda niye λ_i^m ifadeleri elde ettik? Mesela $\mu_1 \lambda_1 v_1$ ifadesi, A ile bir kere carpilinca,

$$\mu_1 \lambda_1 \underbrace{Av_1}_{\lambda_1 v_1} = \mu_1 \lambda_1 \lambda_1 v_1 = \mu_1 \lambda_1^2 v_1$$

olacaktir. Bunu m kere yapinca (1)'in en sagindaki sonucu elde ederiz.

Simdi (1)'in en sagindaki esitligin icinden λ_1^m'i cikartalim (1)'in en solundaki esitlik ile yanyana getirelim,

$$A^m v_o = \lambda_1^m \bigg(\mu_1 v_1 + \mu_n \bigg(\frac{\lambda_2}{\lambda_1}\bigg)^m v_2 + ... + \mu_n \bigg(\frac{\lambda_n}{\lambda_1}\bigg)^m v_n \bigg)$$

Ispatin basinda baskin ozdegerin λ_1 oldugunu soylemistik. O zaman

$$\left|\frac{\lambda_2}{\lambda_1}\right| < 1, ..., \left|\frac{\lambda_n}{\lambda_1}\right| < 1$$

Bu demektir ki limit kosulu $m \to \infty$ durumunda

$$A^m v_o = \lambda_1^m \mu_1 v_1$$

cunku 1'den kucuk olan tum bolumler, katlari alindikca ve o kat (m) cok buyudugunde sifira giderler. Bu bolumleri iceren tum terimler yokolur ve geriye ustteki ifade kalir.

Boylece ust metodunu turetmis olduk. En son ifade sunu soyluyor, herhangi bir vektor v_0'i alalim, ve onu A ile m kere carpalim, ve bu durumda elimize gercek ozvektor v_1'e paralel bir vektor $\lambda_1^m \mu_1 v_1$ gececektir (paralel cunku v_1 degeri tek sayi / skalar $\lambda^m \mu_1$ ile carpilmakta. Bu vektoru normalize ederek bir ozvektor sonucu elde edebiliriz.

Ornek olarak alttaki A'yi alalim, baslangic olarak $v_0 = [1\ 1]$

```
v0 = np.array([1.,1.])
A = np.array([[13., 5], [2,4]])
for i in range(20):
    v0 = np.dot(A,v0)
print 'v0 =',v0
```

```
v0 = [  1.14093076e+23   2.28186151e+22]
```

Sonsuzluk norm'u (infinity norm) ile normalize edersek (sonsuzluk normu bir vektor icindeki en buyuk ogenin alinip bolumde kullanilmasiyla yapilan normalizasyondur),

```
v1 = v0 / np.max(v0)
print 'v1 =', v1
```

```
v1 = [ 1.    0.2]
```

Kontrol edelim,

```
import numpy.linalg as lin
U,D = lin.eig(A)
print D
```

```
[[ 0.98058068 -0.4472136 ]
 [ 0.19611614  0.89442719]]
```

Birinci kolona oldukca yakin bir deger elde ettik, ki en buyuk ozdegere tekabul eden ozvektor orada.

Kaynaklar

`http://www.math.mcgill.ca/feys/documents/tutnotesR18.pdf`

Murphy, K., CS340: Machine Learning Lecture Notes, `www.ugrad.cs.ubc.ca/~cs340`