

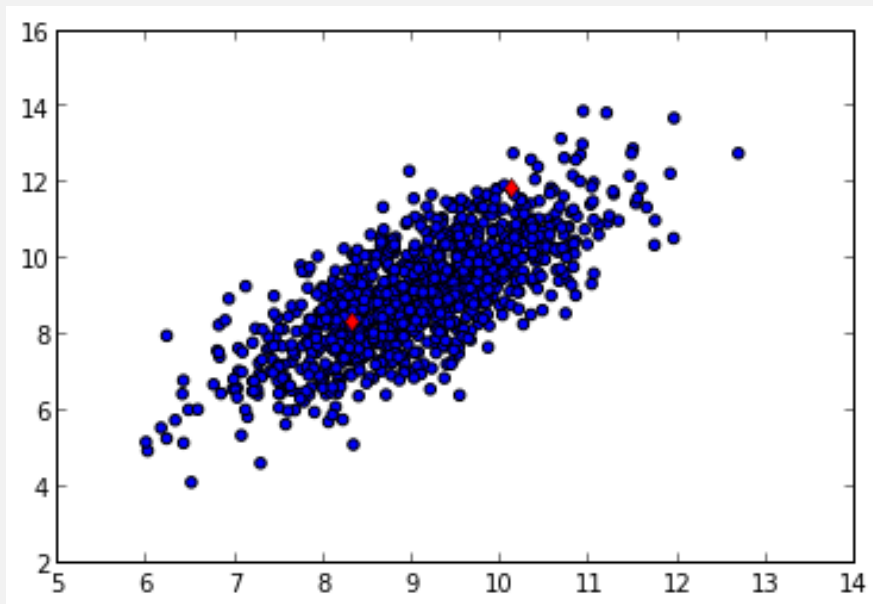
Temel Bileşen Analizi (Principal Component Analysis -PCA-) PCA yöntemi boyut azaltan yöntemlerden biridir, takip edilmeden (unsupervised) işleyebilir. Ana fikir veri noktalarının izdusumunun yapılacağı yonlar bulmaktır ki bu yonlar bağlamında (izdusum sonrası) noktaların arasındaki varyans (variance) en fazla olsun, yani noktalar en “yaygın” şekilde bulunsunlar. Böylece birbirinden daha uzaklaşan noktaların mesela daha rahat kümelenebileceğini umabiliriz. Yani bir diğer amaç hangi değişkenlerin varyansının daha fazla olmasının görülmesi üzerine, o değişkenlerin daha önemli olabileceğinin anlaşılması. Örnek olarak alttaki grafiğe bakalım,

```
from pandas import *  
data = read_csv("testSet.txt",sep="\t",header=None)  
data[:10]
```

	0	1
0	10.235186	11.321997
1	10.122339	11.810993
2	9.190236	8.904943
3	9.306371	9.847394
4	8.330131	8.340352
5	10.152785	10.123532
6	10.408540	10.821986
7	9.003615	10.039206
8	9.534872	10.096991
9	9.498181	10.825446

```
scatter(data.ix[:,0],data.ix[:,1])  
plot(data.ix[1,0],data.ix[1,1], 'rd')  
plot(data.ix[4,0],data.ix[4,1], 'rd')
```

[<matplotlib.lines.Line2D at 0xb52880c>]



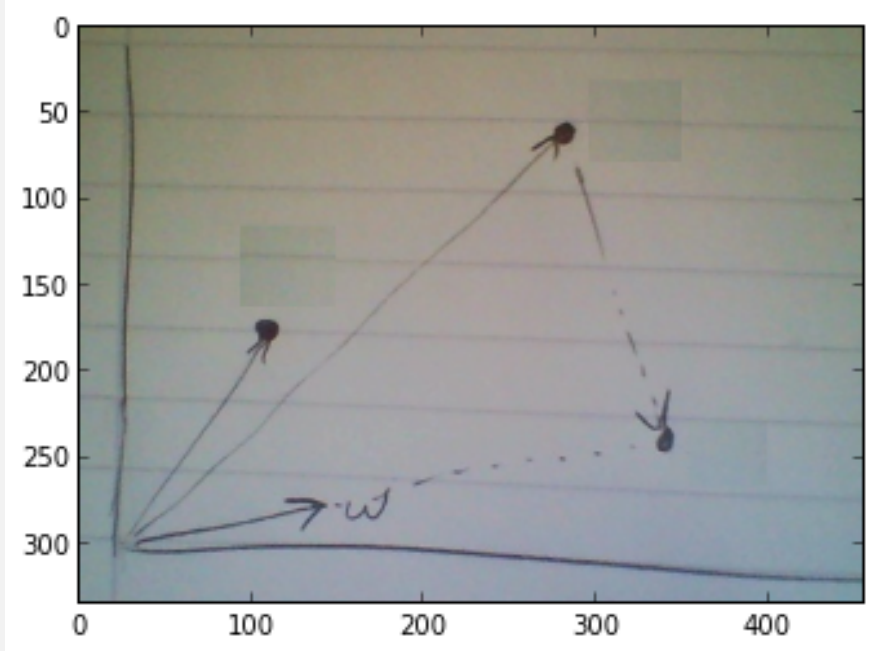
PCA ile yapmaya çalıştığımız öyle bir yon bulmak ki, x veri noktalarının tamamının o yone

izdusumu yapilınca sonuc olacak, “izdusumu yapılmis” z ’nin varyansı en büyük olsun. Bu bir maksimizasyon problemidir. Fakat ondan önce x nedir, z nedir bunlara yakından bakalım.

Veri x ile tüm veri noktaları kastedilir, fakat PCA probleminde genellikle bir “vektorün diğeri üzerine” yapılan izdusumu, “daha optimal bir w yonu bulma”, ve “o yone doğru izdusum yapmak” kelimeleri kullanılır. Demek ki veri noktalarını bir vektor olarak görmeliyiz. Eger üstte kırmızı ile işaretlenen iki noktayı alırsak (bu noktalar verideki 1. ve 4. sıradaki noktalar),

```
img=imread("proj1.png")
plt.imshow(img)
```

<matplotlib.image.AxesImage at 0xa9e5e2c>



gibi bir görüntüden bahsediyoruz. Hayali bir w kullandık, ve noktalardan biri veri noktası, w üzerine izdusum yapılarak yeni bir vektörü / noktayı ortaya çıkartıyor. Genel olarak ifade edersek, bir nokta için

$$z = w^T x$$

Yapmaya çalıştığımız varyansı maksimize etmek demistik. Özel bir izdusum yonunu referans alırsak, en büyük $Var(z_1)$ ’i bulacağız. Not: Bunu söyler söylemez x ’i ve z ’yi rasgele değişkenler olarak gördüğümüzü belirtmiş oluyoruz. Ardı ardına alet çantasından temsili numaraları kullanıyoruz - x ’i bir yandan vektor yapıyoruz, bir yandan bir dağılımdan gelen zar atışı gibi görüyoruz, problemi çözmemize ne yardım edecekse onu sürekli devreye sokuyoruz. Devam edelim, rasgele değişken deyince, demek ki x, z dağılımlardan gelecektir, bu dağılımların çok boyutlu normal (multivariate normal) olduğunu kabul edebiliriz. Peki eğer $x, N(\mu, \Sigma)$ gibi çok boyutlu normal dağılımdan geliyorsa, ki μ ortalama ve Σ kovaryanstır, $w^T x$ nedir?

Bu yeni “seyin” beklentisi ve varyansına bakalım,

$$E[w^T x] = w^T E[x] = w^T \mu$$

$$Var(w^T x) = w^T Var(x) w = w^T \Sigma w$$

Ustteki sonuclarin boyutlarina dikkat: $w^T \mu$ durumunda $1 \times N \cdot N \times 1 = 1 \times 1$, $w^T \Sigma w$ durumunda ise $1 \times N \cdot N \times N \cdot N \times 1 = 1 \times 1$. Iki durumda da tek boyutlu skalar degerler elde ettik. Demek ki w^T ile carpim sonrasi bir $N(w^T \mu, w^T \Sigma w)$ dagilimi elde ederiz ve bu dagilim bir tek boyutlu bir dagilimdir. Yani w yonundeki izdusum bize tek boyutlu bir Gaussian dagilimini verecektir. Bu sonuc aslinda cok sasirtici olmasa gerek, tum veri noktalarini alip, baslangici orijin noktasinda olan vektorlere cevrip ayni yone isaret edecek sekilde duzenliyoruz, bu vektorleri tekrar nokta olarak dusunursek, tabii ki ayni yonu gosteriyorlar, bilahere ayni cizgi uzerindeki noktalara donusuyorlar. Ayni cizgi uzerinde olmak ne demek? Tek boyuta inmis olmak demek.

Bastaki amacimiza donersek, $Var(z_1)$ 'i maksimize etmek ayni anda $Var(w_1^T \Sigma w_1)$ 'i maksimize etmek demektir.

Ufak bir sorun $w_1^T \Sigma w_1$ 'i surekli daha buyuk w_1 'lerle sonsuz kadar buyutebilirsiniz. Bize ek bir kisiltama sarti daha lazim, bu sart $\|w\| = 1$ olabilir, yani w 'nin norm'u 1'den daha buyuk olmasin. Boylece optimizasyon w 'nin buyuklugu uzerinde taklalar atmayacak, sadece yon bulmak ile ilgilenecek, iyi, zaten biz w 'nin yonu ile ilgileniyoruz. Aradigimiz ifadeyi yazalim, ve ek siniri Lagrange ifadesi olarak ekleyelim, ve yeni bir L ortaya cikartalim,

$$L(w_1, \lambda) = w_1^T \Sigma w_1 - \lambda(w_1^T w_1 - 1)$$

Niye eksiden sonraki terim o sekilde eklendi? O terim oyle sekilde secildi ki, $\partial L / \partial \lambda = 0$ alinince $w_1^T w_1 = 1$ geri gelsin / ortaya ciksinsin [2, sf 340], bu Lagrange'in dahice bulusu. Bunu kontrol edebilirsiniz, λ 'ya gore turev alirken w_1 sabit olarak yokolur, parantez icindeki ifadeler kalir ve sifira esitlenince orijinal kisiltama ifadesi geri gelir. Simdi

$$\max_{w_1} L(w_1, \lambda)$$

Turevi w_1 'e gore alirsak, ve sifira esitlersek,

$$2w_1 \Sigma - 2\lambda w_1 = 0$$

$$2w_1 \Sigma = 2\lambda w_1$$

$$\Sigma w_1 = \lambda w_1$$

Ustteki ifade ozdeger, ozvektor ana formulune benzemiyor mu? Evet. Eger w_1 , Σ 'nin ozvektoru ise ve esitligin sagindaki λ ona tekabul eden ozdeger ise, bu esitlik dogru olacaktır.

Peki hangi ozdeger / ozvektor maksimal degeri verir? Unutmayalim, maksimize etmeye calistigimiz sey $w_1^T \Sigma w_1$ idi

Eger $\Sigma w_1 = \lambda w_1$ yerine koyarsak

$$w_1^T \lambda w_1 = \lambda w_1^T w_1 = \lambda$$

Cunku $w_1^T w_1 = 1$ olacagi sartini koymustuk. Neyse, maksimize etmeye calistigimiz deger λ cikti, o zaman en buyuk λ kullanirsak, en maksimal varyansi elde ederiz, bu da en buyuk ozdegerin ta kendisidir.

Demek ki izdusum yapılacak “yon” kovaryans Σ ’nin en büyük özdegerine tekabül eden özvektor olarak secilirse, temel bileşenlerden en önemlisini hemen bulmuş olacağız.

Cebirin geri kalanı w_2, w_3 için devam eder, bu türetmenin detaylarını [1] ve [2] gibi kaynaklarda bulabilirsiniz. Fakat ulaşılan sonuç en bileşenlerin önem sırasının aynen özdeğerlerin büyüklük sırasına tekabül ediyor olması.

Örnek Şimdi tüm bunları bir örnek üzerinde görelim. İki boyutlu örnek veriyi üstte yüklemistik. Şimdi veriyi “sıfırda ortalayacağız” yani her kolon için o kolonun ortalama değerini tüm kolondan çıkartacağız. PCA ile işlem yaparken tüm değerlerin sıfır merkezli olması gerekiyor.

Daha sonra özdeğerlerini, vektörlerini hesaplayabilmek için verinin kovaryansını hesaplayacağız.

```
means = mean(data, axis=0)
meanless_data = data - means
cov_mat = cov(meanless_data, rowvar=0)
eigs,eigv = linalg.eig(mat(cov_mat))
eig_ind = argsort(eigs)
eig_ind
```

```
array([0, 1])
```

Üstteki sonuç sort sonrası elde edilen indis değerleri, sort en küçükten en büyüğe doğru dizimi yapmıştır, en büyük özdeğerin indisi en sondadır. Raslantısal bir şekilde en büyük olan değer en sağda çıktı ama tersi de olabilirdi, neyse, bu özdeğer, vektörleri ekrana basalım,

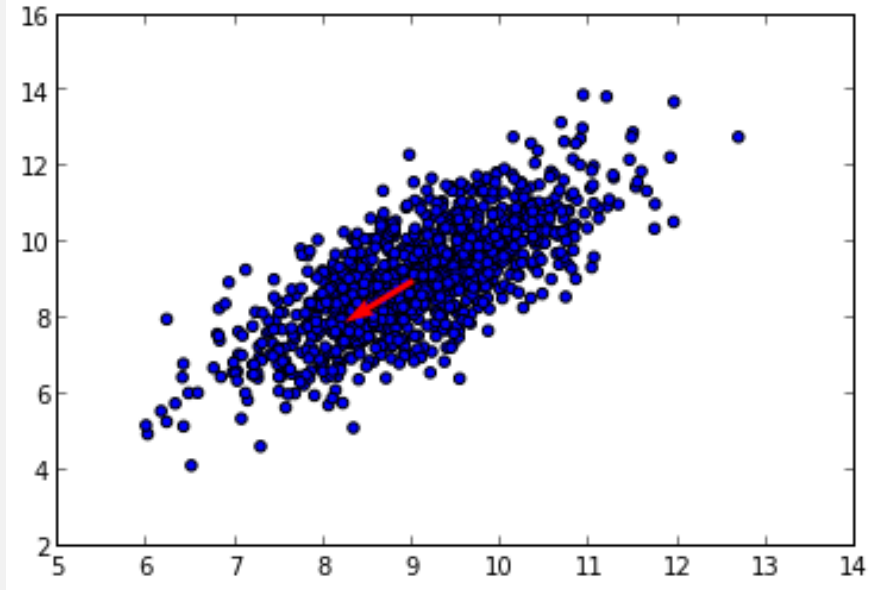
```
print eigs[1],eigv[:,1].T
print eigs[0],eigv[:,0].T
```

```
2.89713495618 [[-0.52045195 -0.85389096]]
0.366513708669 [[-0.85389096  0.52045195]]
```

En büyük olan yonu quiver komutunu kullanarak orijinal veri seti üzerinde gösterelim,

```
scatter(data.ix[:,0],data.ix[:,1])
quiver(9,9,eigv[1,1],eigv[0,1],scale=10,color='r') # merkez 9,9, kabaca secildi
```

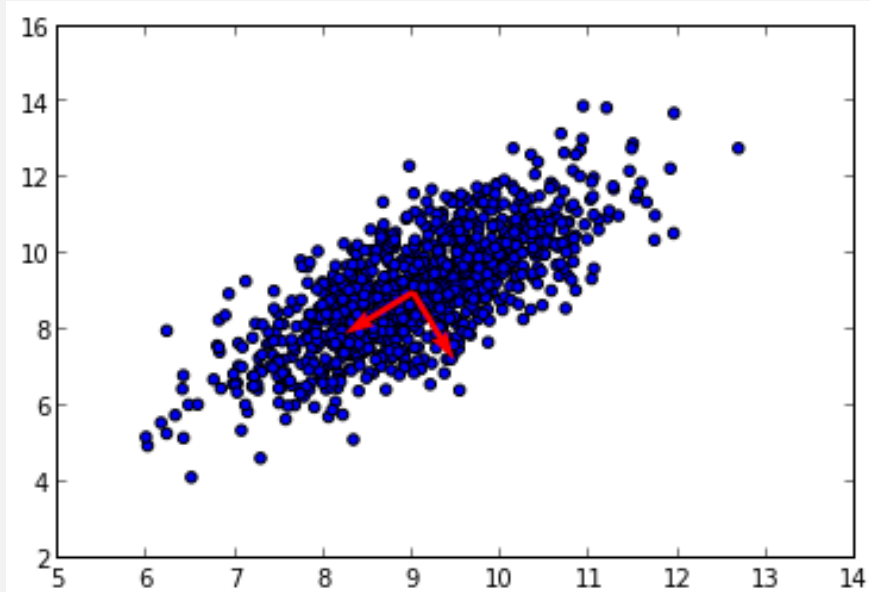
```
<matplotlib.quiver.Quiver at 0xbb6254c>
```



Goruldugu gibi bu yon hakikaten dagilimin, veri noktalarinin en cok yayilmis oldugu yon. Demek ki PCA yontemi dogru sonucu buldu. Her iki yonu de cizersek,

```
scatter(data.ix[:,0],data.ix[:,1])
quiver(9,9,eigv[1,0],eigv[0,0],scale=10,color='r')
quiver(9,9,eigv[1,1],eigv[0,1],scale=10,color='r')
```

<matplotlib.quiver.Quiver at 0xc194c6c>



Bu ikinci yon birinciye dik olmalidi, ve o da bulundu. Aslinda iki boyut olunca baska secenek kalmiyor, 1. yon sonrasi ikincisi baska bir sey olamazdi, fakat cok daha yuksek boyutlarda en cok

yayilimin oldugu ikinci yon de dogru sekilde geri getirilecekti.

[1] Alpaydin, E., Introduction to Machine Learning, 2nd Edition

[2] Strang, G., Linear Algebra and Its Applications, 4th Edition

[3] <http://www.stat.columbia.edu/~fwood/Teaching/w4315/Spring2010/PCA/slides.pdf>