Tasarım ölçümlerinin (metriklerin) istatistiksel analizi
Temel Bileşen Analizi (*Principal Component Analysis PCA*)

- Bir yazılım bileşeni (örneğin sınıf) ile ilgili çok sayıda farklı metrik değeri toplanabilir.
  Örneğin; sınıftaki toplam satır sayısı, kod satırı sayısı (açıklamalar olmadan), metot sayısı, başka sınıflara bağımlılık, testlerde çıkan hata sayısı gibi.
- Farklı metrikler ile elde edilen veriler gerçekten tasarımın farklı özellikleri hakkında bilgiler veriyor mu?
  Hangi metrikler aynı bilgileri, hangileri farklı bilgileri içeriyor?
- Konuyu incelerken aşağıdaki yazıdaki deneylerden yararlanacağız:
  L. C. Briand, J. Wüst, J. W. Daly, and D. Victor Porter, "Exploring the relationships between design measures and software quality in object-oriented systems," *Journal of Systems and Software*, vol. 51, no. 3, pp. 245–273, May 2000.
  - Maryland Üniversitesi Bilgisayar Bilimleri öğrencilerine C++ diliyle program yazdırılmıştır.
  - Orta boyutlarda bir video kiralama yazılımı hazırlanmıştır.
  - Toplam 113 yazılım sınıfı değerlendirilmiştir.
  - Yazılımdaki her sınıf için 67 adet metrik ölçümü yapılmıştır.

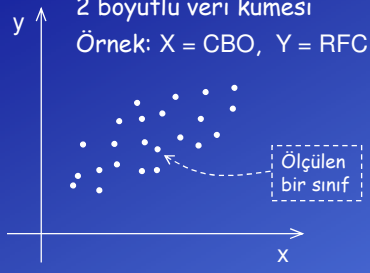
Temel Bileşen Analizi (*Principal Component Analysis PCA*)

- Temel Bileşen Analizi (*Principal Component Analysis - PCA*) yöntemi çok boyutlu (çok değişkenli) bir veri kümesinin karakterini en iyi ortaya koyan temel bileşenleri (birbirine dik vektörleri) bulmaya dayanmaktadır.
- Bu dik vektörlere özvektör (*eigenvector*) adı verilir.
- PCA iki amaçla kullanılmaktadır.
  1. Özvektörlerden daha az bilgi taşıyanları (varyansı küçük olanlar) kaldırarak geride kalan yüksek varyanslı vektörler ile veri kümesini az bilgi kaybıyla yeniden temsil etmek mümkün olmaktadır.
     Böylece n değişkenli bir veri kümesi, eğer p tane özvektör seçilip diğerlerinden vazgeçilirse p boyut ile temsil edilebilir.
  2. Hangi değişkenlerin hangi özvektörlere katkı verdikleri belirlenebilir. Aynı özvektörde yer alan değişkenler benzer bilgiler veriyorlar demektir.
     Aynı bilgileri veren değişkenlerden sadece uygun olanlar modele koyulabilir.

Temel Bileşen Analizi (*Principal Component Analysis PCA*) devamı

2 boyutlu veri kümesi

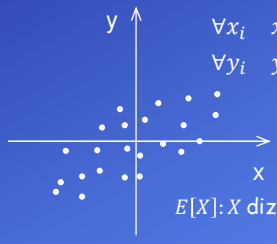
Örnek: X = CBO, Y = RFC olabilir



Standardize veri kümesi

$\forall x_i \quad x_i - E[X]$

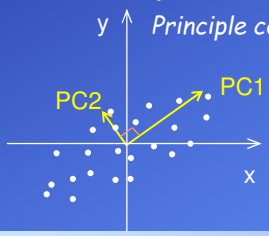
$\forall y_i \quad y_i - E[Y]$



$E[X]$: X dizisinin ortalaması

Temel bileşenler, Özvektörler

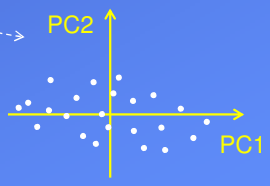
Principle components PC



Döndürme

PC1 ve PC2 veri kümesi hakkında farklı bilgiler içeriyorlar.
Vektörler dik (*orthogonal*)

Verinin temel bileşenlere göre temsil edilmesi



Temel Bileşen Analizi (*Principal Component Analysis PCA*) devamı

Boyutun azaltılması:

- n boyutlu veri kümesinin n adet temel bileşeni olur.
- Bunlardan daha az bilgi taşıyanlar kaldırılarak veri kümesinin boyutu küçültülebilir.

Örnek:

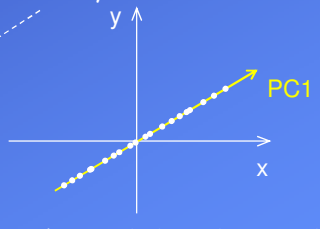
İki boyutlu örnek şekilde PC1 daha çok bilgi taşımaktadır bu nedenle PC2 kaldırılarak veri sadece PC1 ile ifade edilecektir.

Verinin sadece PC1'e göre temsil edilmesi:

PC1

PC2'nin sağladığı bilgi kaybolmuştur.

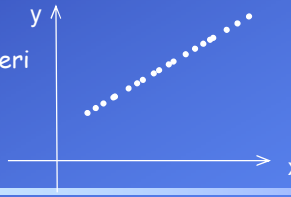
1 boyutlu standardize veri kümesi



PC2'nin sağladığı bilgi kaybolmuştur.

Orijinal veri kümesine dönebilmek için ortalamalar veriye eklenmeli:

1 boyuttan elde edilen orijinal veri kümesi



Temel bileşen analizinin metrik değerlendirmesinde kullanımı:

- Yazılımda veri kümesini her sınıf için farklı metriklerle toplanan değerler oluşturur.
- İncelenen örnek çalışmada (Briand et.al.) değişkenler her sınıf için toplanan metrik değerleridir.
- Her sınıf için 67 farklı metrik değeri alınmıştır. (Soru: Hangileri aynı/farklı?)
  Buna göre toplanan veri kümesi 67 boyutludur.
- İncelenen öğrenci programlarındaki toplam 113 sınıf değerlendirilmiştir.
  Demek ki her boyutta 113 değer vardır. (113 elemanlı, 67 tane dizi)
- Bu veri kümesine PCA uygulandığında varyans değeri yüksek 16 adet temel bileşen (PC) belirlenmiştir.
- Metrikleri değerlendirebilmek için temel bileşenleri yorumlamak gerekir.
  Bulunan bileşenler (PC) üzerinde hangi değişkenlerin daha etkili olduğunu bulmak için bileşen döndürme (*component rotation*) yöntemi kullanılmaktadır.
- Böylece aşağıdaki bilgiler elde edilebilir:
  - Hangi temel bileşen (ağırlıklı olarak) hangi metriklerden oluşmaktadır?
  - Aynı temel bileşende bulunan metrikler yazılımın belli bir boyutu (örneğin bağımlılık) hakkında birlikte bilgi vermektedirler.
  - Farklı temel bileşenlerde bulunan metrikler farklı bilgiler vermektedirler.
  - Seçilen temel bileşenlerde yer almayan metrikler yazılım hakkında belirleyici bilgi vermemektedirler.

Dikkat: PCA yöntemi sadece üzerinde inceleme yapılan veri kümesine göre sonuç verir. Bu sonuçları tek bir kümeye bakarak genellemek mümkün değildir.

Temel Bileşen Analizi (*Principal Component Analysis PCA*) devamı

Matematiksel İşlemler:

1. n boyutlu veri kümesindeki veriler her boyutta standardize hale getirilir.
   Her boyutun ortalaması o boyuttaki verilerden çıkartılır.

$$xs_i^d = x_i^d - E[X^d]$$

xs_i^d: d. boyuttaki i. standardize veri

x_i^d: d. boyuttaki i. orijinal veri

$E[X^d]$: d. boyuttaki verilerin ortalaması

Bizim konumuzda her boyut (n adet değişken) bir metriğe karşı gelmektedir.
Örneğin CBO boyutunda (dizisinde) tüm sınıfların CBO değerleri yer almaktadır.
Kümedeki varlıklar ise (m adet) sınıflardır.

2. Tüm boyutlar (değişkenler) arasında ikili olarak kovaryans değerleri hesaplanır.

Hatırlatma:

$$cov(X,Y) = \frac{\sum_{i=1}^{m}(x_i - E[X])(y_i - E[Y])}{m}$$

$$cov(X,Y) = cov(Y,X)$$

Kovaryans iki değişken arasındaki ilişkiyi gösterir.

$cov(X,Y)$ pozitifse X ve Y değişkenlerinin birlikte arttığı, negatifse biri artarken diğerinin azaldığı anlaşılır.

Eğer $cov(X,Y) = 0$ ise iki değişken arasında bir ilişki olmadığı anlaşılır.

Tüm boyutlar arasında ikili olarak kovaryans değeri hesaplanır.

Standardize verilerin ortalaması sıfır olduğundan

$$C(p,r) = cov(XS^p, XS^r) = \frac{\sum_{i=1}^{m}(xs_i^p)(xs_i^r)}{m}, \qquad p = 1,2,..,n \;;\; r = 1,2,...,n$$

XS^p, XS^r: p. ve r. boyutlardaki (dizilerdeki) standardize veriler.

Kovaryans değerleri ile nxn boyutundaki simetrik kovaryans matrisi C^{nxn} oluşturulur.

Bu matrisin (i,j). elemanı i. değişken ile j. değişken arasındaki kovaryanstır.

3. Kovaryans matrisinin (C^{nxn}) özvektörleri (*eigenvector*) E^{nx1}bulunur.

$$C^{nxn} \times E^{nx1} = \lambda \times E^{nx1}$$

λ: Özvektörün özdeğeridir (*eigenvalue*).

nxn boyutunda bir matrisin n adet özvektörü (ve özdeğeri) olur.

Bu özvektörler veri kümesinin karakterini gösteren doğruları (temel bileşenleri) belirleyen vektörlerdir.

Özvektörler birbirlerine diktirler (*orthogonal*).

Özdeğer o vektördeki varyans (değişimle) bağlantılıdır.

Örnek:

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 2 \end{bmatrix} = 4 \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

Bu örnekte $\begin{bmatrix} 3 \\ 2 \end{bmatrix}$ vektörü, $\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}$ matrisinin özvektörlerinden biridir.

Bu özvektörün özdeğeri $\lambda = 4$ tür.

Özvektörler boyutlarına bölünerek birim vektör (boyu 1) haline getirilirler.

Örneğin, $\begin{bmatrix} 3 \\ 2 \end{bmatrix}$ vektörünü boyutu $\sqrt{3^2 + 2^2} = \sqrt{13}$ tür.

Vektörün her iki elemanı da $\sqrt{13}$ 'e bölünerek birim vektör elde edilir.

4. Özvektörler, özdeğerlerine (λ) göre büyükten küçüğe doğru sıralanırlar.

Özdeğeri daha büyük olan vektör veri kümesi hakkında daha çok bilgi taşımaktadır.

Eğer veri kümesinin boyutunun küçültülmesi isteniyorsa özdeğeri küçük olan özvektörler atılabilirler.

Özvektörler oluşturulan sıralarına göre sütunlara dizilerek veri kümesinin özellik vektörü (*feature vector*) FV^{nxn} oluşturulur.

$$FV^{nxn} = E_1^{nx1} E_2^{nx1} \dots E_n^{nx1} \qquad \lambda_1 > \lambda_2 > \lambda_3 \dots$$

Bu matris standardize veri kümesiyle çarpılarak veriler temel bileşenlere göre temsil edilebilir.

$$XPC^{nxm} = TFV^{nxn} \times XS^{nxm}$$

XS^{nxm} : Ortalamaya göre standardize edilmiş n boyutlu veri kümesidir.

Matrisin satırlarında değişik boyutlar (yazılım örneğinde farklı metrikler) yer alır.

Sütunlarda ise üzerinde ölçüm yapılan m tane varlık (örnekte sınıflar) yer alır.

XPC^{nxm} : Temel bileşenlere göre ifade edilen veri kümesi.

TFV^{nxn} : Transpoze edilmiş özellik vektörü. Özdeğer vektörleri satırlardadır.

5. Eğer istenirse veri kümesinin boyutu küçültülebilir.

Özdeğeri küçük olan özvektörler veri kümesi hakkında daha az bilgi taşıdıklarından özdeğeri belli bir değerden daha küçük olan özvektörler özellik vektöründen çıkartılabilirler.

Eğer n tane özvektörün sadece p tanesi seçilirse özellik vektörü (*feature vector*) FV^{nxp} aşağıdaki gibi oluşur.

$$FV^{nxp} = E_1^{nx1} E_2^{nx1} \dots E_p^{nx1} \qquad \lambda_1 > \lambda_2 > \dots > \lambda_p > seçilen\ bir\ eşik\ değeri$$

Bu matris standardize veri kümesiyle çarpılarak veriler p adet temel bileşene göre p boyutlu olarak göre temsil edilebilir.

$$XPC^{pxm} = TFV^{pxn} \times XS^{nxm}$$

XS^{nxm} : Ortalamaya göre standardize edilmiş n boyutlu veri kümesidir.

XPC^{pxm} : Temel bileşenlere göre ifade edilen p boyutlu veri kümesi.

TFV^{pxn} : Transpoze edilmiş p boyutlu özellik vektörü. Özdeğer vektörleri satırlardadır.

Bu işlemler tersine yapılarak p boyutlu orijinal veri kümesi kayıplı olarak elde edilir.

Temel bileşen analizinin örnek deneysel (*empirical*) çalışmada kullanılması:

- Üçer öğrenciden oluşan 8 grup.
- Orta boyutta video kiralama ve müşteri yönetimi yazılımı.
- Unix, C++, GNU, OSF/MOTIF kütüphanesi kullanılıyor.
- Bütün yazılımlardan toplam 113 sınıf değerlendiriliyor. Her sınıf için 67 ayrı metrik değeri elde ediliyor.
  Aslında 83 metrik var, ancak varyansı küçük olanlar ve 6 tane sıfırdan farklı değer üretmeyenler eleniyor. (Metrikler yazının ekinde yer almaktadır)
- Bağımlılık metrikleri iki ayrı şekilde ölçülmüştür:
  a. Arşiv sınıflarına bağımlılık, b. Diğer (programa ait) sınıflara bağımlılık
- Profesyonel yazılımcılar bu sınıfları inceleyerek hataları belirliyorlar.

Çalışmada iki konu ele alınıyor:

1. Metrikler birbirinden bağımsız mı? Veri hakkında belirleyici bilgiyi hangi metrikler veriyor?
2. Metrikler ile sınıflarda hata bulunması arasında bir bağlantı var mıdır?

Birinci sorunun yanıtını bulmak üzere veriler PCA yöntemi ile değerlendirilmiştir.
Bu bölümde PCA yöntemiyle elde edilen sonuçlar ele alınacaktır.

Örnek çalışmada metrik değerlerinin temel bileşen analizi ile incelenmesi:

- Özdeğerleri 1.0'dan büyük olan ($\lambda > 1.0$) özvektörler temel bileşen olarak seçilmiştir.
- Böylece 16 adet temel bileşen (PC) elde edilmiştir.
  Bu 16 PC veri kümesindeki varyansın %88.6'sını kapsamaktadır.
  Buradan çıkan sonuç; ölçümlerde büyük miktarda fazlalık vardır.
- Döndürme (*rotated components, varimax rotation*) uygulanarak değişkenlerin temel bileşen üzerindeki etkileri belirlenmiştir.
  Temel bileşen üzerindeki ağırlığı 0.7'den büyük olan değişkenler (metrikler) seçilmiştir.
  Böylece o temel bileşenin yazılımın hangi özelliğine denk düştüğü yorumlanmaya çalışılmıştır.
- Yorumların sonucunda 16 bileşenden 13 tanesi anlamlı bulunmuştur.

Elde edilen temel bileşenler ve yorumlarına ilişkin örnekler:

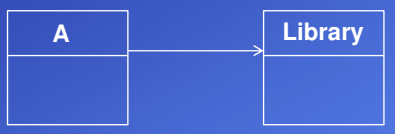
PC1: Etkili değişkenler MPC_L, ICP_L, NIH_ICP_L, OMMIC_L

Sonunda "_L" olan metrikler kütüphane sınıfları ile olan ilişkileri ifade eder.

Bu bileşende adı geçen metrikler bir sınıfın kütüphane sınıflarına mesaj çağırıları üzerinden olan bağımlılıklarını ifade eder (Bkz. ilgili yazıdaki metrik tablosu).

Örneğin MPC_L (*Message Passing Coupling*) incelenen sınıfın, kütüphane sınıflarının kaç tane metodunu çağırdığı ile ilgilidir.

Bir sınıfın başka sınıfların metotlarını çağırması "*import coupling*" olarak adlandırılır.



Import coupling of A from Library.

Yorum:

Sonuç olarak bu bileşen bir sınıfın kütüphane sınıflarına metot çağırılarıyla oluşan bağımlılığını ifade eder.

Burada adı geçen metrikler aynı bileşende yer aldıkları için benzer bilgiler verirler (benzer şekilde değişirler).

PC2: Etkili değişkenler CBO, CBO', RFC_1, RFC_∞, MPC, ICP, NIH_ICP, OMMIC.

Bu bileşende adı geçen metrikler bir sınıfın kütüphane sınıfı olmayan diğer program sınıflarına mesaj çağırıları üzerinden olan bağımlılıklarını ifade eder (Bkz. ilgili yazıdaki metrik tablosu).

Yorumlar:

PC1 ve PC2 karşılaştırıldığında, kütüphane sınıflarına olan bağımlılıklarla diğer sınıflara olan bağımlılıklarla ilgili metriklerin farklı temel bileşenlerde yer aldığı görülür.

Buna göre bu iki farklı bağımlılık türü birbiri ile ilişkili değildir.

Kütüphane sınıflarına çok bağımlı olan bir sınıf diğer sınıflara çok bağımlı olamayabilir veya diğer sınıflara çok bağımlı olan bir sınıf kütüphane sınıflarına bağımlı olmayabilir.

Hatırlatma:

PCA yöntemi incelenen veri kümesine bağlı sonuçlar verir.

Başka bir yazılım grubunda başka sonuçlar elde edilebilir.

PC3: Etkili değişkenler LCOM1, LCOM2, LCOM3, LCOM4, NMA, NumPara

LCOMx metrikleri bir sınıftaki metotların eriştikleri niteliklerin ortak olup olmamasına göre sınıfın uyumunu ölçmeye çalışırlar.

LCOMx metrikleri normalize olmadıklarından üst sınırları yoktur.

NMA, NumPara ise boyutla ilgili metriklerdir.

NMA (*number of methods added*): Alt sınıfa eklenen metot sayısı.
NumPara (*number of parameters*): Sınıfta gerçeklenen metotların toplam parametre sayısı.

Yorumlar:

- LCOM4 diğer LCOM metriklerinden farklı olarak iki metodun birbirini çağırmasını da uyum açısından hesaba katmaktadır.
  Ancak bu farklılık bu metriğin farklı bir boyut yaratmasına (farklı bilgiler vermesine) neden olamamıştır.
- Boyut metrikleri uyum ile ilgili bilgiler vermektedir.
  Bir sınıftaki metot sayısı ve buna bağlı olarak parametre sayısı artarsa ortak nitelik kullanan metotların sayısı (uyum) azalır.

PC4: Etkili değişkenler LCOM5, Coh, Co, LCC, TCC

Bu metrikler normalizedir; alt ve üst sınır değerleri vardır.

LCOM5 uyumsuzluğu gösterir.
LCOM5 = 0: en yüksek uyum; LCOM5 = 2: en düşük uyum;

Diğer metrikler uyumla orantılıdır; yüksek değer yüksek uyum anlamına gelir.

Bu nedenle LCOM5'in PC4'teki katsayısı negatif çıkmıştır.

Yorumlar:

- Normalize olmayan ve normalize olan uyum metriklerinin ayrı temel bileşenlerde yer alması bunların farklı bilgiler taşıdığını gösterir.
- Uyumu hangi grup ile ifade etmek gerekir, ikisi de birlikte kullanılabilir mi araştırılması gereken bir konudur.

Genel Yorumlar:

- Elde edilen 16 temel bileşenden 3 tanesi anlamlı şekilde yorumlanamamıştır.
- Yorumlanan 13 temel bileşenden 9 tanesi bağımlılıkla ilgilidir.
- Bir bileşen (PC4) sadece uyum metriklerinden oluşmaktadır.
- İki bileşen (PC8, PC9) sadece kalıtımla ilgilidir.
- Az sayıda bileşen farklı ölçümlerin karışımından oluşmaktadır.

  Örneğin; PC3 uyum ve boyutla, PC7 bağımlılık ve uyum, PC8 kalıtım ve kalıtım bağımlılığı ile ilgilidir.

  Bu nedenle bu ölçümlerin (metriklerin) farklı bilgiler verdiği söylenebilir.
- Bazı metrikler hiçbir temel bileşende yer almamıştır (ağırlıkları düşüktür).
- Bazı metrikler birden fazla bileşende yer almıştır.

  Örneğin CBO. Bu metrik hem dışa bağımlılığı hem de dışarıdan bağımlılığı ölçer.

  RFCx hem bağımlılık hem de boyutla ilgilidir.

Genel Yorumlar (devamı):

- Zaman içinde bazı temel metriklerin türevleri yaratılmıştır.

  Örneğin RFC_1, RFC_∞ ; LCOMx ; TCC ve LCC gibi.

  Yeni metrikler oluşturmaktaki amaç eskilerinin bazı eksikliklerini gidermek ve daha "doğru" sonuçlar almaktır.

  Ancak yapılan analiz metriklerin türevlerinin aynı bileşenlerde yer aldığını göstermiştir.

  Demek ki bu "geliştirilmiş" metrikler de aynı bilgiyi vermektedir.
- Bağımlılık ile ilgili çok sayıda temel bileşen olduğuna göre sınıfların bağımlılığını tek bir metrikle ölçmek ve değerlendirmek mümkün değildir.
- Normalize olan ve olmayan uyum metrikleri (örneğin TCC ve LCOM) farklı bileşenlerde yer alıyorlar.

  Demek ki metrik değerini normalize etmek (örneğin metot sayısına oranlamak) metriğin verdiği bilgiyi değiştiriyor.