Redução de dimensionalidade

Jacques Wainer

IC – Unicamp

Outubro 2015

Por que reduzir a dimensionalidade dos dados?

- em modelos preditivos, muitos atributos podem confundir o classificador/regressor - redução como uma etapa *necessária* do processo.
- em modelos exploratórios e preditivos, muitos atributos podem causar o problema da "maldição da dimensionalidade" (*curse of dimensionality*) que torna a noção de distancia entre pontos do espaço essencialmente inútil - e portanto não ideias baseadas em distancia podem não funcionar
- Meus dados podem ser visto como a combinação de poucos padrões? Quais são estes padrões?

Remoção de atributos

Normalmente, para dados com muitos atributos:

- pode-se remover um de 2 atributos que são altamente relacionados entre si (correlação para atributos numéricos, informação mutua para atributos categóricos): os dois atributos dizem praticamente a mesma coisa
- pode -se remover atributos categóricos que tem uma distribuição de dados muito (muito!!) desbalanceada - poucos dados de um tipo e muitos de outro.
- pode-se remover atributos com pequena variância (em relação a atributos similares) - isso inclui o exemplo acima!
- remover atributos categóricos com muitos valores presentes (CEP) - mas agrupa-los pode ser util (os primeiros 2 dígitos do CEP).

Mas se voce esta buscando anomalias, então os itens anteriores pode ser a chave de encontrar anomalias!

Para modelos exploratórios, na maioria da vezes remoção de atributos não é algo interessante.

Remoção de atributos

Para modelos preditivos

- remover atributos que são pouco relacionados com o atributo de saída:
- pode ser derivado de considerações teóricas - lateralidade para prever a altura de uma pessoa
- mas pode ser empiricamente definido dos proprios dados:

Para modelos modelos preditivos é quase sempre bom remover alguns (ou muitos) atributos.

Principal Component Analisys - PCA

Para dados vetoriais:

- determinar as direções que explicam/contém a maioria da variação dos dados
- as direções são chamadas de *principal components*, e ordenadas de tal forma que o PC1 é a direção de maior variação, PC2 é a segunda maior variação, etc. ▸ link e ▸ link
- os PC são ortogonais entre si

Intuições sobre PCA

- PCA modela os dados como elipsóides (da dimensão dos dados) e o 1o componente é a direção mais alongada do elipsóide, o 2o PC a segunda direção mais alongada

PCA no R

Ha duas funções que calcula o PCA no R padrão (e mais em outros pacotes)

- as 2 funções usam algoritmos diferentes, um é "melhor" que o outro
- a função `prcomp` ▸ link é a implementação mais moderna e deve ser usada sempre que possível
- `princomp` ▸ link é a implementação mais antiga e não preferencial

PCA no R

```
> data(USArrests)
> pca1=prcomp(USArrests, scale. = T)
> pca1
Standard deviations:
[1] 1.5748783 0.9948694 0.5971291 0.4164494

Rotation:
                PC1        PC2        PC3         PC4
Murder   -0.5358995  0.4181809 -0.3412327  0.64922780
Assault  -0.5831836  0.1879856 -0.2681484 -0.74340748
UrbanPop -0.2781909 -0.8728062 -0.3780158  0.13387773
Rape     -0.5434321 -0.1673186  0.8177779  0.08902432
```

A função `prcomp` deve ser chamado com `scale. = T` quase sempre!

PCA no R

- o PCA primeiro padroniza os dados. Isto é:
- remove a média de cada atributo dos dados
- divide cada atributo pelo desvio padrão dos valores do atributo (é isso que o `scale. = T` faz).
- a parte da saída `Standard deviations:` indica o desvio padrão dos dados nas direções PC1 PC2 etc.
- a parte `Rotation:` indica as direções dos PC1, PC2 etc (em função das coordenadas originais (`Murder`, `Assault`, etc

PCA no R

A parte da padronização que divide pelo desvio padrão pode ser problemática se algum atributo tem um valor só. Neste caso o desvio padrão é 0 e dividir por 0 da erro. É possível não usar o `scale. = T`

```
> pca2=prcomp(USArrests)
> pca2
Standard deviations:
[1] 83.732400 14.212402  6.489426  2.482790

Rotation:
                PC1         PC2         PC3         PC4
Murder   0.04170432 -0.04482166  0.07989066 -0.99492173
Assault  0.99522128 -0.05876003 -0.06756974  0.03893830
UrbanPop 0.04633575  0.97685748 -0.20054629 -0.05816914
Rape     0.07515550  0.20071807  0.97408059  0.07232502
```

PCA no R

princomp da os mesmos resultados, mas a chamada é um pouco diferente

```
> pca3=princomp(USArrests,cor=T)
> pca3
Call:
princomp(x = USArrests, cor = T)

Standard deviations:
   Comp.1    Comp.2    Comp.3    Comp.4
1.5748783 0.9948694 0.5971291 0.4164494

 4  variables and  50 observations.
> pca3$loadings

Loadings:
         Comp.1 Comp.2 Comp.3 Comp.4
Murder   -0.536  0.418 -0.341  0.649
Assault  -0.583  0.188 -0.268 -0.743
UrbanPop -0.278 -0.873 -0.378  0.134
Rape     -0.543 -0.167  0.818
```

PCA e redução de dimensionalidade

Como o PCA determina os componentes principais por order de variância, a forma mais comum de reduzir a dimensionalidade é pegar apenas os primeiros PC dos dados.

O atributo `x` do resultado do PCA tem os valores dos dados nas coordenadas PC. É so usar as primeiras coordenadas

PCA e redução de dimensionalidade

```
> head(pca1$x)
                PC1        PC2         PC3          PC4
Alabama    -0.9756604  1.1220012 -0.43980366  0.154696581
Alaska     -1.9305379  1.0624269  2.01950027 -0.434175454
Arizona    -1.7454429 -0.7384595  0.05423025 -0.826264240
Arkansas    0.1399989  1.1085423  0.11342217 -0.180973554
California -2.4986128 -1.5274267  0.59254100 -0.338559240
Colorado   -1.4993407 -0.9776297  1.08400162  0.001450164

> head(pca1$x[,1:2])
                PC1        PC2
Alabama    -0.9756604  1.1220012
Alaska     -1.9305379  1.0624269
Arizona    -1.7454429 -0.7384595
Arkansas    0.1399989  1.1085423
California -2.4986128 -1.5274267
Colorado   -1.4993407 -0.9776297
```

PCA e redução de dimensionalidade

Pode-se trabalhar direto nos dados `pca$x[,1:2]` ou pode-se converter os dados de volta para as dimensões originais

```
> z = pca1$x[,1:2] %*% t(pca1$rotation[,1:2])
> head(z)
               Murder   Assault   UrbanPop        Rape
Alabama    0.9920554 0.7799093 -0.7078698  0.3424735
Alaska     1.4788608 1.3255791 -0.3902348  0.8713524
Arizona    0.6265723 0.8790939  1.1300983  1.0720877
Arkansas   0.3885458 0.1267449 -1.0064890 -0.2615597
California 0.7002647 1.1700159  2.0282388  1.6133934
Colorado   0.3946699 0.6906107  1.2703841  0.9783655
```

Os resultados estão nas coordenadas originais mas refletem dados que estão em um subespaço de 2 dimensões (nas coordenadas PC). Note que os dados transformados estão padronizados (media $= 0$ e desvio padrão $= 1$).

Como usar o PCA para redução de dimensionalidade

Automático

- voce já sabe que quer manter apenas k dimensões dos dados
- Faça `pca1$x[,1:k]`

Como usar o PCA para redução de dimensionalidade

KDD:

- Critério de Kaiser - fique com os PC cuja desvio padrão são maiores que 1

```
> pca1
Standard deviations:
[1] 1.5748783 0.9948694 0.5971291 0.4164494
```

- neste caso ficariamos com as 2 primeiras dimensões
- fique com os PCs cuja soma cumulativa dos quadrados do desvio padrão atinge de 80 a 90% da variância total

```
> cumsum(pca1$sdev^2)/sum(pca1$sdev^2)
[1] 0.6200604 0.8675017 0.9566425 1.0000000
```

- neste caso ficariamos com 2 ou 3 dimensões
- busque um "joelho" (ou "cotovelo") no *scree plot* (diagrama do quadrado do `sdev` pelo PC ▸ link
- este ultimo método é menos confiável.

Scale ou não scale?

- usar o `scale. = T` permite que se use as regras acima para escolher o número de dimensões
- mas o `scale` modifica as distancias entre os pontos e portanto analises que se baseiam em distancia (quase todas) ficam modificadas.
- mas `scale` é a "unica" forma de padronizar atributos que tem medidas/unidades diferentes

Recomendação:

- jogue fora atributos que não variam
- use o `scale` principalmente se os atributos são numéricos e tem diferentes "sentidos"/unidades
- se todos so atributos são categóricos, e forma convertidos usando o one-hot encoding, normalmente não se faz o `scale`

Redução de dimensionalidade como aproximação de matrizes

- Os dados podem ser vistos como uma matrix X de n linhas (n dados) e d dimensões (os dados tem d atributos numéricos)
- vamos aproximar X pelo produto de 2 matrizes DN e B onde
- ND (novos dados) tem n linhas mas k colunas ($k < d$ - dai a redução de dimensionalidade)
- B (bases) é uma matrix de k linhas e d colunas.
- ND representa os dados usando uma nova base (como no PCA) e precisa de menos coordenadas nessa nova base - esta base tem k coordenadas
- e B é a nova base - cada linha de B descreve uma nova coordenada (em função das d coordenadas originais)
- ▸ link é um boa ilustração disso mas ele troca linhas por colunas mais aidante nos slides (quem é coordenada e quem é dado).
- descubra ND e B de tal forma que o produto $ND \times B$ seja o mais proximo de X possível.

Redução de dimensionalidade como aproximação de matrizes

- Pode-se colocar restrições na matriz ND (normalmente).
- Por exemplo, os valores de ND devem ser positivos ou 0 (NNMF). Desta forma cada novo dado é uma soma ponderada dos padrões de B - isto é uma das alternativas para sistemas de recomendação
- se X é entre 0 e 1, tanto ND quanto B podem ser limitados para 0 e 1, que da uma interpretação probabilistica - (Latent Dirichelet Allocation) LDA é uma das formas mais comuns de redução de dimensionalidade para texto.
- ND pode ter muitos 0 e os outros valores positivos. Neste caso a solução é chamada *esparsa*
- algumas formas de agrupamento (próxima aula) podem ser vistos como fatoração. Cada linha de ND pode ter todos os valores 0 e apenas um 1. Nesse caso ND indica a que grupo o dado pertence, e B são os "representantes" de cada grupo. Há variações esparsas para agrupamento com interseção

PCA como aproximação de matrizes

- PCA é a fatoração SVD (singular value decomposition) da matrix $X = U\Sigma V^T$ onde Σ é uma matriz $k \times k$ diagonal, e U e V são ortogonais.
- PCA é a forma de decompor $X = ND \times B$ que possui menor erro
- PCA também pode ser visto como a decomposição em autovalores e autovalores da matrix de covariância de X ($X \times X^T$)

Tarefa

- Usando as 4 primeiras dimensões do dataset `iris` (que ja vem dentro do R), salve no arquivo iris2d.csv o resultado reduzir o numero de dimensões de iris para 2. Entrega até o fim do dia.
- Utilizando as respostas do questionário para os alunos
  - remover os atributos com texto livre
  - converter os atributos ordenados ("Proficiência em X" e "Dominio em X") para números (de 1 a 5)
  - descobrir qual o número de dimensões deve-se manter do dataset
  - Entrega até o 28/10