Universidade Federal de São Carlos
Aprendizado de Máquina
Prof. Tiago A. Almeida <talmeida@ufscar.br>

Exercício Prático 6: Análise de Componentes Principais

Introdução

Neste exercício você implementará o método de análise de componentes principais (PCA) e verá como ele pode ser eficientemente empregado para comprimir dados. Antes de começar este exercício, é recomendável que você revise os conceitos apresentados em aula.

Arquivos incluídos neste exercício

`ex06.m` – Script geral do exercício
`ex06Data.mat` – Base de dados com 50 amostras representadas por 2 atributos cada
`ex06Faces.mat` – Base de dados de imagens faciais. Possui 5000 amostras com 1024 atributos, cada qual sendo uma imagem 32×32.
`plotarLinha.m` – Plota uma linha no gráfico
`exibirImagem.m` – Plota dados 2D em um grid
`normalizarAtributos.m` – Função para normalizar as amostras
[$\star$] `pca.m` – Realiza a análise de componentes principais
[$\star$] `projetarDados.m` – Projeta os dados de entrada em uma dimensão reduzida
[$\star$] `recuperarDados.m` – Computa aproximação dos dados originais a partir dos dados projetados.

$\star$ indica os arquivos que você precisará completar.

O arquivo `ex06.m` conduzirá todo o processo desse exercício.

Análise de Componentes Principais

Neste exercício, você usará o PCA para reduzir a dimensionalidade das amostras de entrada. Inicialmente, será empregada uma base de dados pequena com apenas duas dimensões para auxiliar na implementação do método. Posteriormente, uma base maior com 5000 amostras de imagens faciais ilustrará o potencial da técnica.

Para ajudá-lo a entender como o PCA funciona, você iniciará a implementação usando um conjunto de dados 2D ilustrados na Figura 1. Nesta parte do exercício, você irá visualizar o que ocorre quando o PCA é usado para reduzir os dados de 2D para 1D. Na prática, você pode querer reduzir os dados de 10.000 para 1.000 dimensões.

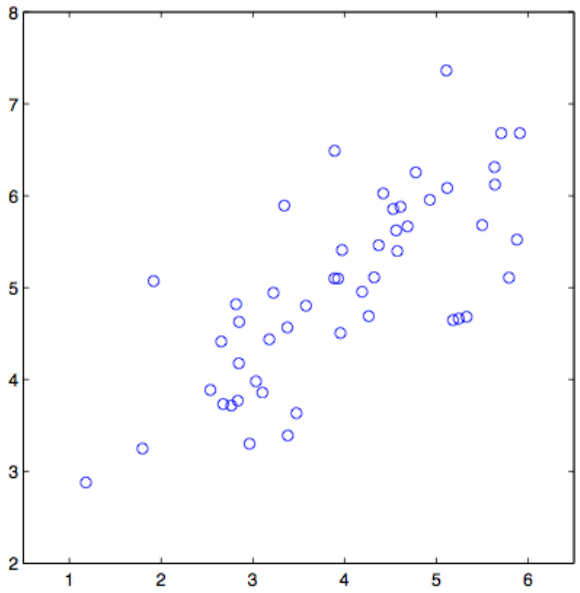

Figura 1: Visualização dos dados

Implementação do PCA

Nesta parte do exercício, você precisará implementar o PCA. Basicamente, ele consiste de dois passos: o primeiro, é calcular a matriz covariância dos dados. Para isso, você poderá usar a função `SVD` para calcular os autovetores $U_1, U_2, \ldots, U_n$. Estes correspondem aos componentes principais da variação dos dados.

Antes de empregar o PCA, é importante primeiro normalizar os dados. No *script* fornecido, a normalização é realizada pela função `normalizarAtributos`. Após a normalização, todos as atributos passam a ter média zero e desvio padrão igual à um.

Após a normalização, o PCA pode ser usado para computar os componentes principais. Sua tarefa é completar o código `pca.m` para calcular os componentes principais do conjunto de dados de entrada. Em primeiro lugar, é necessário calcular a matriz covariância dos dados, que é dada por:

$$\Sigma = \frac{1}{m} X^T X$$

sendo X a matriz de dados com amostras em linhas, e m a quantidade de exemplos. Note que Σ é uma matriz $n \times n$.

Após o cálculo da matriz de covariância, você pode usar a função SVD para computar os componentes principais. A função SVD retornará a matriz U com os componentes principais de Σ e a matriz S com os autovalores de Σ em uma matriz diagonal.

Após a aplicação do PCA no conjunto de dados, serão plotados os componentes principais encontrados (Figura 2). Neste ponto, será impresso o primeiro componente principal (autovetor) encontrado. É esperado que ele seja aproximadamente igual à $[-0,707 - 0,707]$. Observe que é possível que tenham sinais opostos, uma vez que U_1 e $-U_1$ são igualmente válidos.

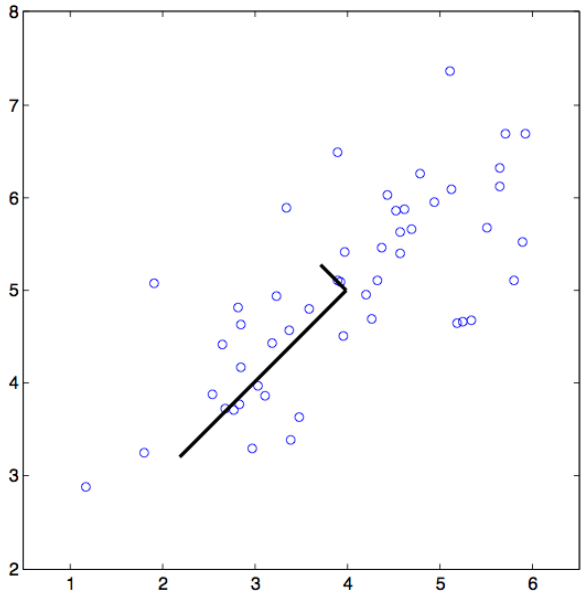

Figura 2: Autovetores extraídos da base de dados

Você precisará completar a função ***pca.m****.*

Redução de dimensionalidade usando o PCA

Após computar os componentes principais, você pode usá-los para reduzir a dimensão dos atributos do conjunto de dados, projetando cada exemplo em um espaço dimensional menor, $x^{(i)} \rightarrow z^{(i)}$ (por exemplo, projetando os dados de 2D para 1D). Nesta parte do exercício, você precisará usar os autovetores retornados pelo PCA e projetar as amostras em um espaço de uma dimensão.

Na prática, se você estivesse usando um algoritmo de aprendizagem, você poderia usar os dados projetados ao invés dos dados originais. Com isso, você pode treinar seu modelo mais rápido, pois há bem menos dimensões nos dados de entrada.

Projetando os dados sobre os componentes principais

Neste ponto, você precisará completar a função `projetarDados.m`. Especificamente, dados o conjunto de dados X, os componentes principais U e o número de dimensões K. É possível projetar cada amostra de X sobre os primeiros K componentes de U. Observe que os K componentes principais são dados pelas primeiras K.

É esperado que a projeção da primeira amostra ($Z^(1)$) sobre a primeira dimensão retorne valor aproximadamente igual à 1,481 (ou possivelmente -1,481).

Você precisará completar a função ***pca.m****.*

Reconstrução de uma aproximação dos dados

Após projetar os dados em um espaço dimensional menor, é possível recuperar uma aproximação projetando-os de volta para o espaço dimensional original. Sua tarefa é completar a função `reconstruirDados.m` para projetar cada amostra de Z de volta para o espaço original e retornar a aproximação reconstruída em X_{rec}.

É esperado que a primeira amostra recuperada seja aproximadamente igual à $[-1,047 - 1,047]$.

Você precisará completar a função ***pca.m****.*

Visualizando as projeções e exibindo valores

Após completar todas as funções, o *script* irá plotar tantos os dados originais normalizados quanto os reconstruídos (Figura 3). Os dados originais normalizados são representados por círculos azuis, enquanto que os dados projetados são representados por círculos vermelhos. A projeção efetivamente só mantém as informações na direção dada por U_1.

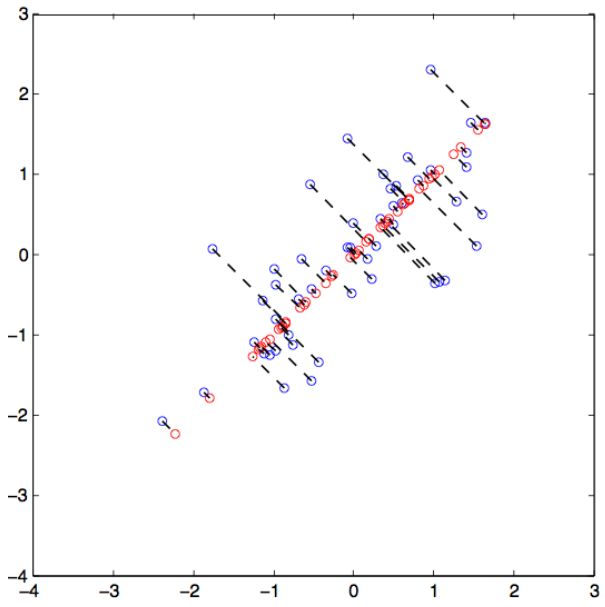

Figura 3: Amostras originais normalizadas e projetadas após o PCA

Finalmente, é exibida uma tabela com os valores originais de cada amostra (x), o valor projetado (z) e a amostra reconstruída (x_{rec}).

Base de dados de imagens faciais

Nesta parte do *script*, o PCA será aplicado para comprimir amostras de imagens faciais presentes na base `ex05Faces.mat`. O conjunto de dados contém 5000 amostras representadas por imagens 32×32 em escala de cinza (vetor linha com 1024 atributos). Neste ponto, as 100 primeiras amostras serão exibidas.

Aplicação do PCA

Inicialmente, os dados são normalizados e, a seguir, o PCA é executado. É importante observar que cada componente principal em U (cada linha) é um vetor de comprimento n (sendo que para o conjunto de dados faciais, $n = 1024$). Esses componentes podem ser visualizados pela remodelação de cada um deles usando a matriz 32×32, correspondente aos pixels do conjunto de dados original. Em seguida, os primeiros 36 componentes principais são apresentados. Neste ponto, é possível alterar o código para exibir mais componentes principais para ver como eles capturam os detalhes.

Redução de dimensionalidade

Uma vez que os componentes principais foram computados, eles podem ser empregados para reduzir a dimensionalidade do conjunto de dados. Isso permite que algum algoritmo de aprendizado seja aplicado sob uma entrada de tamanho menor (por exemplo, 100 dimensões) ao invés da original de 1024 dimensões.

Em seguida, os dados originais serão projetados sobre os 100 primeiros componentes principais. Portanto, cada imagem passará a ser representada por um vetor $z^{(i)} \in \Re^{100}$.

Para visualizar o efeito da redução da dimensionalidade, uma reconstrução aproximada dos dados é realizada e as imagens originais e projetadas são exibidas lado a lado. A partir da reconstrução, é possível observar que a estrutura geral e aparência (contornos) das faces são mantidas, enquanto que os detalhes são perdidos. Tal redução (mais que $10\times$) do conjunto de dados pode ajudar a acelerar o algoritmo de aprendizado de maneira expressiva. Por exemplo, se você estava treinando o método de regressão logística para realizar o reconhecimento facial, é possível usar os dados de dimensão reduzida com apenas 100 dimensões ao invés da matriz de pixels original.