Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina

Análise de Componentes Principais

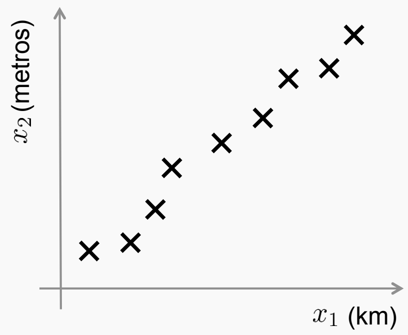
Prof. Tiago A. Almeida

Motivação

- Aplicação para compressão de dados
  - Redução de espaço
    - economia de espaço para armazenamento
    - redução de tempo para transmissão
    - redução da influência de ruídos ou *outliers*
    - **visualização dos dados (3D, 2D)**
    - **aceleração do aprendizado dos algoritmos de AM**

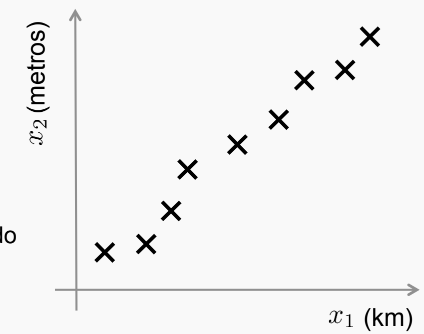
Compressão de dados

- **Exemplo**
- Redundância nos dados
  - Muito comum na prática
  - Dados coletados de diferentes fontes (ex: Internet)
  - Elevado número de atributos
  - Difícil detecção manual



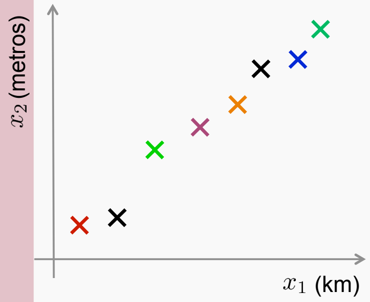
Compressão de dados

- **Exemplo**
- Redundância nos dados
  - Muito comum na prática
  - Dados coletados de diferentes fontes (ex: Internet)
  - Elevado número de atributos
  - Difícil detecção manual
- Um dos eixos pode ser removido
- 2D → 1D



Compressão de dados

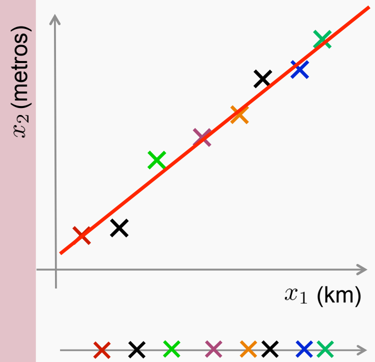
- 2D $\longrightarrow$ 1D



Tiago A. Almeida

Compressão de dados

- 2D $\longrightarrow$ 1D

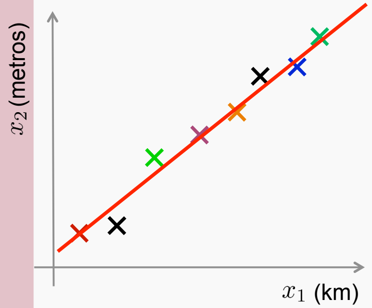


Projeção dos dados na linha vermelha

Tiago A. Almeida

Compressão de dados

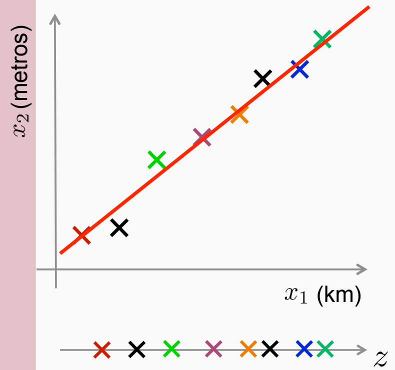
- 2D $\longrightarrow$ 1D



Tiago A. Almeida

Compressão de dados

- 2D $\longrightarrow$ 1D

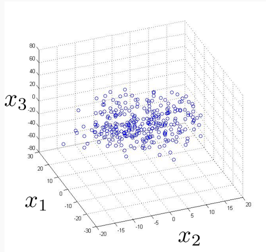


$$x^{(1)} \in \Re^2 \quad \rightarrow z^{(1)} \in \Re$$
$$x^{(2)} \in \Re^2 \quad \rightarrow z^{(2)} \in \Re$$
$$\vdots$$
$$x^{(m)} \in \Re^2 \quad \rightarrow z^{(m)} \in \Re$$

Novo atributo z que representa (uma aproximação) x_1 e x_2

Compressão de dados

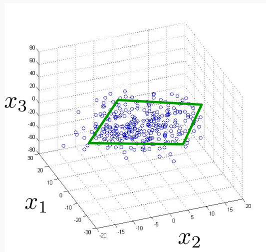
- 3D → 2D



$x^{(i)} \in \Re^3$

Compressão de dados

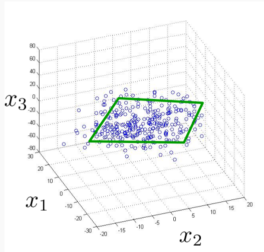
- 3D → 2D



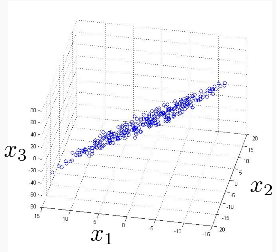
$x^{(i)} \in \Re^3$

Compressão de dados

- 3D → 2D



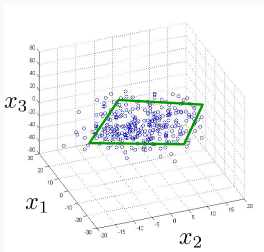
$x^{(i)} \in \Re^3$



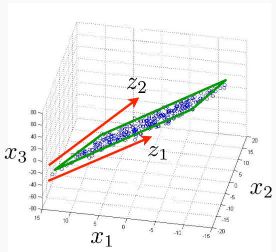
Dados originais são projetados no plano

Compressão de dados

- 3D → 2D



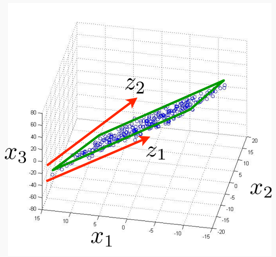
$x^{(i)} \in \Re^3$

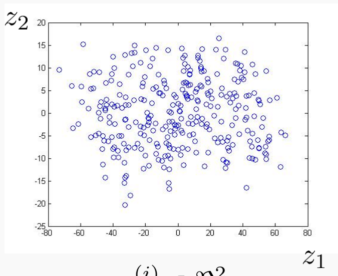


Dados originais são projetados no plano

Compressão de dados

- 3D $\rightarrow$ 2D

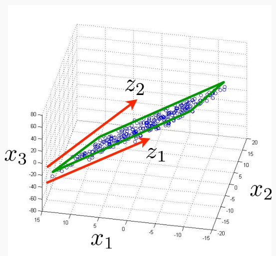


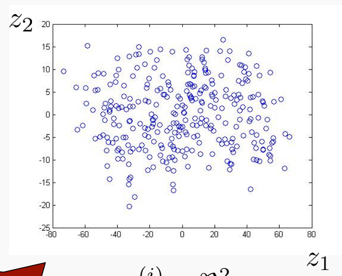


$z^{(i)} \in \Re^2$

Compressão de dados

- 3D $\rightarrow$ 2D





$z^{(i)} \in \Re^2$

Aplicações reais:
1000D $\rightarrow$ 100D

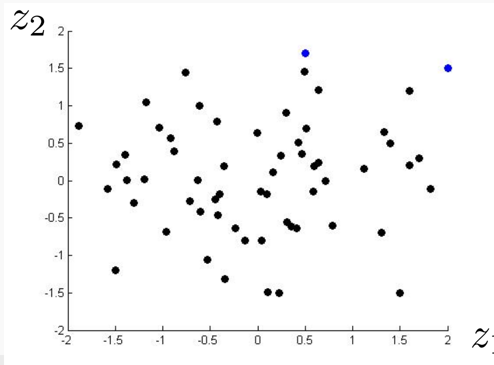
Visualização de dados

- Aplicações reais normalmente trabalham com bases de dados com muitas amostras formadas por muitos atributos. Ex: $X \in \Re^{100}$
  - É impossível visualizar os dados no formato original

- Muitas vezes a visualização dos dados auxilia na escolha dos métodos e na detecção de *outliers*
  - Deseja-se reduzir as dimensões para plotar

Visualização de dados

- Mapeamento de $X \in \Re^{100} \rightarrow Z \in \Re^2$



Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina

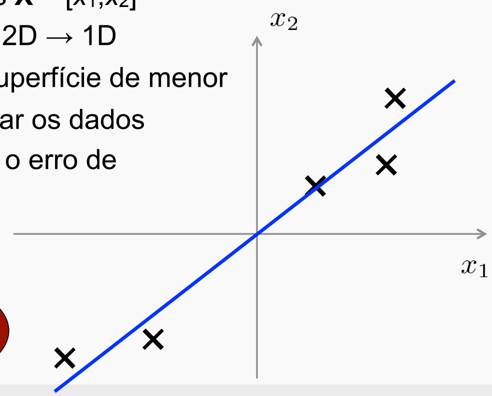
Análise de Componentes Principais

Método

Análise de componentes principais: Ideia

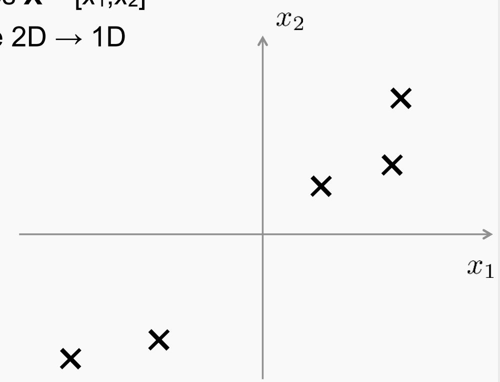
- Seja a base de dados **X** = [x_1,x_2]
- Deseja-se reduzir de 2D → 1D
- Objetivo: encontrar superfície de menor dimensão para projetar os dados de forma a minimizar o erro de projeção ortogonal

Aconselhável aplicar normalização nos dados antes de aplicar PCA

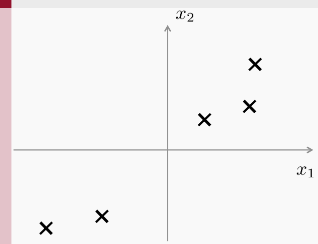


Análise de componentes principais: Ideia

- Seja a base de dados **X** = [x_1,x_2]
- Deseja-se reduzir de 2D → 1D

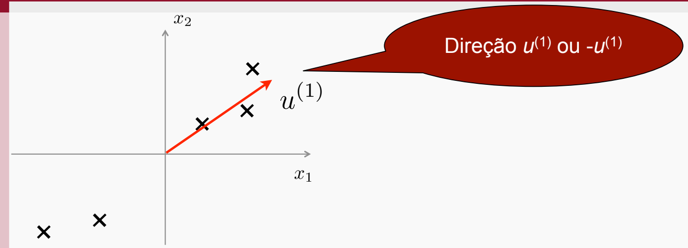


Análise de componentes principais: Ideia



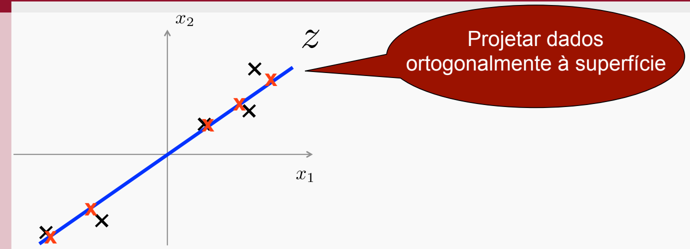
- Reduzir de 2D → 1D: encontrar direção (vetor $u^{(1)} \in \Re^2$) para criar projeção que minimize o erro de projeção

Análise de componentes principais: Ideia



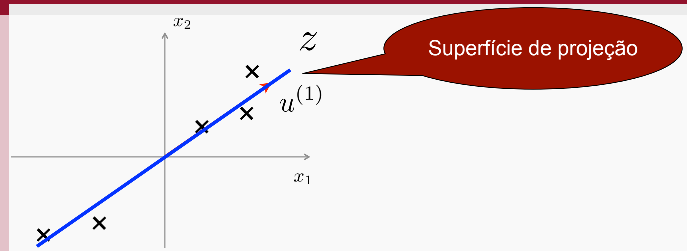
- Reduzir de 2D → 1D: encontrar direção (vetor $u^{(1)} \in \Re^2$) para criar projeção que minimize o erro de projeção

Análise de componentes principais: Ideia



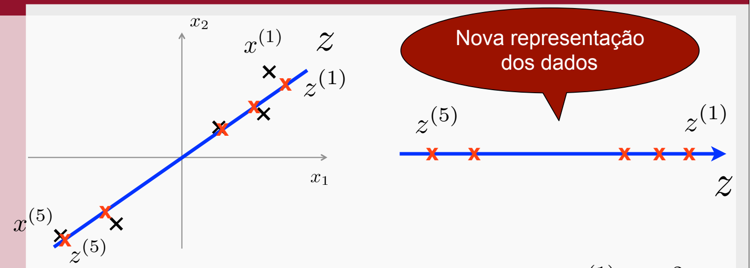
- Reduzir de 2D → 1D: encontrar direção (vetor $u^{(1)} \in \Re^2$) para criar projeção que minimize o erro de projeção

Análise de componentes principais: Ideia



- Reduzir de 2D → 1D: encontrar direção (vetor $u^{(1)} \in \Re^2$) para criar projeção que minimize o erro de projeção

Análise de componentes principais: Ideia



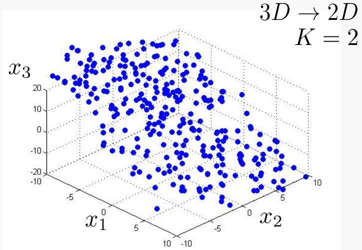
- Reduzir de 2D → 1D: encontrar direção (vetor $u^{(1)} \in \Re^2$) para criar projeção que minimize o erro de projeção

Análise de componentes principais: Ideia

- **Ideia geral**
  - Reduzir de ***n*D** → ***k*D**: encontrar ***k*** vetores $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$ para criar projeção que minimize o erro de projeção

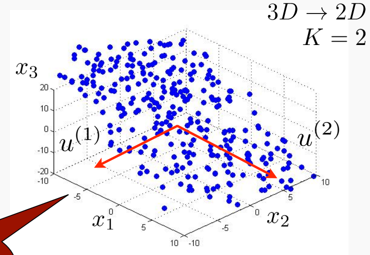
Análise de componentes principais: Ideia

- **Ideia geral**
  - Reduzir de ***n*D** → ***k*D**: encontrar ***k*** vetores $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$ para criar projeção que minimize o erro de projeção
- **Exemplo**



Análise de componentes principais: Ideia

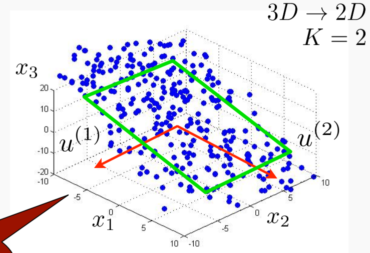
- **Ideia geral**
  - Reduzir de ***n*D** → ***k*D**: encontrar ***k*** vetores $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$ para criar projeção que minimize o erro de projeção
- **Exemplo**



Direções

Análise de componentes principais: Ideia

- **Ideia geral**
  - Reduzir de ***n*D** → ***k*D**: encontrar ***k*** vetores $u^{(1)}, u^{(2)}, \ldots, u^{(k)}$ para criar projeção que minimize o erro de projeção
- **Exemplo**



Superfície de projeção

Análise de componentes principais: Etapa 1

- **Pré-processamento dos dados**
  - Seja a base de treinamento: $x^{(1)}, x^{(2)}, \ldots, x^{(m)}$
  - Aplicar normalização por padronização:

$$\mu_j = \frac{1}{m}\sum_{i=1}^{m} x_j^{(i)} \qquad \sigma_j = \sqrt{\frac{1}{m-1}\sum_{i=1}^{m}(x_i - \mu_j)^2}$$

  - Substituir cada $x_j^{(i)}$ por $x_j^{(i)} = \dfrac{x_j^{(i)} - \mu_j}{\sigma_j}$

Análise de componentes principais: Algoritmo

- Reduzir dimensionalidade de *n* para *k*
  - Computar matriz de covariância Σ:

$$\Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})^T$$

  - Computar autovetores de $\Sigma \in \Re^{nxn}$:

```
[U, S, V] = svd(Sigma);
```

$U, S, V \in \Re^{n \times n}$

Análise de componentes principais: Algoritmo

- `[U, S, V] = svd(Sigma):`

$$U = \begin{bmatrix} | & | & & | \\ u^{(1)} & u^{(2)} & \ldots & u^{(n)} \\ | & | & & | \end{bmatrix} \in \mathbb{R}^{n \times n}$$

Direções

Análise de componentes principais: Algoritmo

- `[U, S, V] = svd(Sigma):`

$$U = \begin{bmatrix} \underbrace{\begin{matrix} | & | & \\ u^{(1)} & u^{(2)} & \ldots \\ | & | & \end{matrix}}_{k} & \begin{matrix} | \\ u^{(n)} \\ | \end{matrix} \end{bmatrix} \in \mathbb{R}^{n \times n}$$

$$Ur \in \Re^{n \times k}$$

Selecionar *k* primeiras direções

Análise de componentes principais: Algoritmo

- Tem-se que:

$$x \in \Re^{n \times 1}$$
$$Ur \in \Re^{n \times k}$$

Cada amostra do conjunto de treinamento

- Deseja-se:

$$x \in \Re^{n \times 1} \rightarrow z \in \Re^{k \times 1}$$

Análise de componentes principais: Algoritmo

- Tem-se que:

$$x \in \Re^{n \times 1}$$
$$Ur \in \Re^{n \times k}$$

k direções selecionadas

- Deseja-se:

$$x \in \Re^{n \times 1} \rightarrow z \in \Re^{k \times 1}$$

Análise de componentes principais: Algoritmo

- Tem-se que:

$$x \in \Re^{n \times 1}$$
$$Ur \in \Re^{n \times k}$$

Projeções de *x* em *Ur*

- Deseja-se:

$$x \in \Re^{n \times 1} \rightarrow z \in \Re^{k \times 1}$$

Análise de componentes principais: Algoritmo

- Tem-se que:

$$x \in \Re^{n \times 1}$$
$$Ur \in \Re^{n \times k}$$

Projeções de *x* em *Ur*

- Deseja-se:

$$x \in \Re^{n \times 1} \rightarrow z \in \Re^{k \times 1}$$

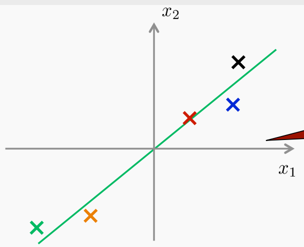
$$\boxed{z = Ur^T x}$$

Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina

Análise de Componentes Principais

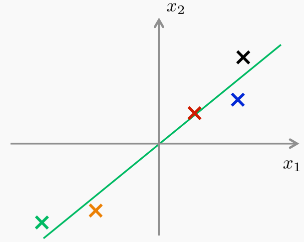
Reconstrução

Análise de componentes principais: Reconstrução do dado



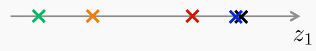
Dado original

Análise de componentes principais: Reconstrução do dado

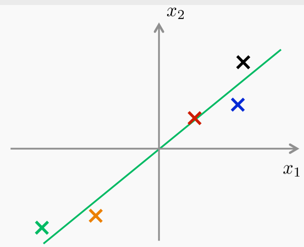


$$z = Ur^T x$$

Dado projetado



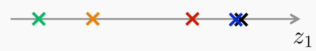
Análise de componentes principais: Reconstrução do dado



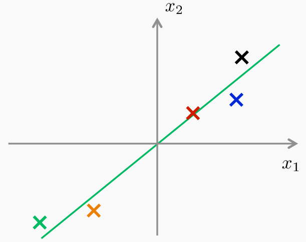
Deseja-se:

$$z \in \Re \rightarrow x \in \Re^2$$

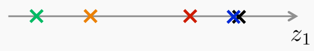
$$z = Ur^T x$$



Análise de componentes principais: Reconstrução do dado



$$z = Ur^T x$$



Deseja-se:
$z \in \Re \rightarrow x \in \Re^2$

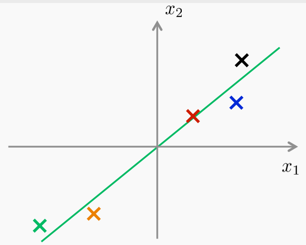
Como:
$Ur \in \Re^{n \times k}$
$z \in \Re^{k \times 1}$

Então:
$x_{approx} = Ur.z$

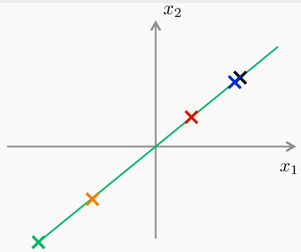
Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina

Análise de Componentes Principais
Parâmetros

Análise de componentes principais: Reconstrução do dado



$$z = Ur^T x$$



$$x_{approx} = Ur.z$$

Análise de componentes principais: Escolha de k

- ***k*** = número de dimensões de cada amostra (em PCA: quantidade de componentes principais)
- Como escolher um bom valor para ***k***?

Análise de componentes principais: Escolha de k

- EQM da projeção: $\frac{1}{m}\sum_{i=1}^{m} ||x^{(i)} - x_{approx}^{(i)}||^2$
- Variância total dos dados: $\frac{1}{m}\sum_{i=1}^{m} ||x^{(i)}||^2$
- Normalmente, *k* escolhido é o menor valor para

$$\frac{\frac{1}{m}\sum_{i=1}^{m} \|x^{(i)} - x_{approx}^{(i)}\|^2}{\frac{1}{m}\sum_{i=1}^{m} \|x^{(i)}\|^2} \leq 0.01 \qquad (1\%)$$

99% da variância é mantida

Análise de componentes principais: Escolha de k - SVD

- `[U, S, V] = svd(Sigma)`
- **S**: matriz diagonal *n* x *n* com autovalores (S_{11}, S_{22}, S_{nn})

$$\frac{\frac{1}{m}\sum_{i=1}^{m} ||x^{(i)} - x_{approx}^{(i)}||^2}{\frac{1}{m}\sum_{i=1}^{m} ||x^{(i)}||^2} = 1 - \frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \leq 0.01$$

Análise de componentes principais: Escolha de k

- Procedimento para escolha de *k*:
  - **Início**
  - Executar PCA com *k* = 1
  - Computar U_r, $z^{(1)}$, $z^{(2)}$, ..., $z^{(m)}$, $x_{approx}^{(1)}$, ... $x_{approx}^{(m)}$
  - Checar se

$$\frac{\frac{1}{m}\sum_{i=1}^{m} \|x^{(i)} - x_{approx}^{(i)}\|^2}{\frac{1}{m}\sum_{i=1}^{m} \|x^{(i)}\|^2} \leq 0.01$$

  - Caso contrário, *k* = *k* + 1 e volta para **início**

Análise de componentes principais: Escolha de k - SVD

- `[U, S, V] = svd(Sigma)`
- **S**: matriz diagonal *n* x *n* com autovalores (S_{11}, S_{22}, S_{nn})
- Escolher menor *k* de tal forma que:

$$\frac{\sum_{i=1}^{k} S_{ii}}{\sum_{i=1}^{n} S_{ii}} \geq 0.99$$

Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina

Análise de Componentes Principais

Aplicação

Análise de componentes principais: Aplicando PCA

- Seja o conjunto de treinamento $X \in \Re^{10.000}$

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$$

Análise de componentes principais: Aplicando PCA

- Seja o conjunto de treinamento $X \in \Re^{10.000}$

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$$

Supor que k =1000 preserva 95% da variância dos dados

Análise de componentes principais: Aplicando PCA

- Seja o conjunto de treinamento $X \in \Re^{10.000}$

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$$

- Extrair apenas os atributos:

$$x^{(1)}, x^{(2)}, \ldots, x^{(m)} \in \mathbb{R}^{10000}$$

Análise de componentes principais: Aplicando PCA

- Seja o conjunto de treinamento $X \in \Re^{10.000}$

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})$$

- Extrair apenas os atributos:

$$x^{(1)}, x^{(2)}, \ldots, x^{(m)} \in \mathbb{R}^{10000}$$

$$\downarrow PCA$$

$$z^{(1)}, z^{(2)}, \ldots, z^{(m)} \in \mathbb{R}^{1000}$$

Análise de componentes principais: Aplicando PCA

- Novo conjunto de treinamento $Z \in \Re^{1000}$

$$(z^{(1)}, y^{(1)}), (z^{(2)}, y^{(2)}), \ldots, (z^{(m)}, y^{(m)})$$

Análise de componentes principais: Aplicando PCA

- Novo conjunto de treinamento $Z \in \Re^{1000}$

$$(z^{(1)}, y^{(1)}), (z^{(2)}, y^{(2)}), \ldots, (z^{(m)}, y^{(m)})$$

- Mapear amostras de teste $x^{(i)} \rightarrow z^{(i)}$ usando os parâmetros (U_r) encontrados na base de treinamento ($z = U_r^T.x$) e computar $h(z)$