APRENDIZAGEM DE MÁQUINA
(usando Python)

Thiago Marzagão

LSA, LDA

problema de trabalhar c/ textos: dimensionalidade

- A quantidade de colunas cresce rapidamente com a quantidade de documentos
- Quantidade de colunas pode chegar facilmente à casa dos milhões.
- Isso resulta em problemas computacionais.
- Duas soluções:
- ... redução de dimensionalidade (LSA)
- ... extração de tópicos (LDA, HDP, outros)

Latent Semantic Analysis - LSA

- Objetivo aqui é simplesmente reduzir a quantidade de colunas.
- Não há qualquer premissa sobre o processo de geração dos dados (DGP).

Latent Semantic Analysis - LSA

- Matematicamente LSA é apenas o nome que damos a um algoritmo de decomposição de matrizes - singular value decomposition (SVD) - quando aplicado a textos.
- SVD pode ser aplicado a qualquer tipo de dados, não apenas textos.
- LSA é às vezes chamada de LSI (Latent Semantic Indexing).

Latent Semantic Analysis - LSA

- Primeiro passo: escolher número de dimensões - k.
- Como escolher k?
- Let the data speak!

Latent Semantic Analysis - LSA

- Primeiro passo: escolher número de dimensões - k.
- Como escolher k?
- Let the data speak!
- "Mas Thiago, isso é ateorético!"
- Sim!
- Lembre-se da distinção entre aprendizagem de máquina e econometria.
- Foco aqui é na PREDIÇÃO.

Latent Semantic Analysis - LSA

- Segundo passo: decompor a matriz TF-IDF (vamos chamá-la de A), usando Singular Value Decomposition (SVD):
- $\underset{m \times n}{A} = \underset{m \times m}{U} \underset{m \times n}{\Sigma} \underset{n \times n}{V^*}$
- U é uma matriz ortogonal (i.e., $U'U = I$) cujas colunas são os vetores singulares à esquerda de A.
- Σ é uma matriz diagonal cujos valores não-zero são os valores singulares de A,
- V^* é uma matriz ortogonal cujas colunas são os vetores singulares à direita de A.
- (V^* é a matriz transposta conjugada de V)
- SVD é apenas um dentre vários métodos de decomposicção de matrizes (Cholesky, QR, etc).
- A solução p/ SVD é encontrada *iterativamente*, por tentativa e erro.

Latent Semantic Analysis - LSA

- Terceiro passo: truncar U, Σ, V^*
- Mantemos apenas as k primeiras colunas de U ($\tilde{U}$)
- ... as k primeira linhas e k primeiras colunas de Σ ($\tilde{\Sigma}$)
- ... e as k primeiras linhas de V^* ($\tilde{V^*}$)
- $\tilde{U}$ mapeia palavras a tópicos: $\tilde{u}_{ij}$ é o peso da palavra i no tópico j
- $\tilde{\Sigma}\tilde{V^*} = \tilde{S}$ mapeia tópicos a documentos: $\tilde{s}_{ij}$ é o peso do tópico i no documento j

Latent Semantic Analysis - LSA

- Os tópicos são ordenados.
- O primeiro tópico - i.e., a primeira coluna de $\tilde{S}$ - captura mais variação que o segundo.
- O segundo mais que o terceiro.
- E assim por diante.
- Portanto, se extraímos $k = 300$ e depois $k = 200$ os 200 primeiros tópicos serão os mesmos nos dois casos.
- Cada tópico é independente de todos os tópicos extraídos depois.
- É isso: $\tilde{S}$ é a matriz com dimensionalidade reduzida, que vamos usar no lugar da matriz TF-IDF.

Latent Dirichlet Allocation - LDA

- LSA é flexível: não assume nada sobre o processo de geração dos dados.
- (LSA não é estatística, é matemática.)
- Problema: difícil interpretar os coeficientes.
- O que exatamente é o "peso" da palavra i no tópico j?
- O que exatamente é o "peso" do tópico i no documento j?
- Não há uma interpretação natural p/ "peso" aqui.

Latent Dirichlet Allocation - LDA

- LDA, por outro lado, assume que as palavras são geradas de uma determinada maneira.
- Perde-se generalidade, mas ganha-se interpretabilidade (mais sobre isso daqui a pouco).

Latent Dirichlet Allocation - LDA

- C/ LDA, não fazemos a transformação TF-IDF.
- LDA modela a contagem bruta de palavras, não de suas transformações.

Latent Dirichlet Allocation - LDA

- O modelo teórico:
- Passo 1: escolher N, o número de palavras no documento.
- $N \sim Poisson(\xi)$
- Passo 2: criar um vetor k-dimensional, θ, com as proporções dos tópicos no documento.
- Ex.: $\theta = [0.3, 0.2, 0.5]$
- $\theta \sim Dir(\alpha)$
- Passo 3: sorteamos cada palavra, w.
- P/ cada palavra primeiro sorteamos o tópico, z, dos k tópicos que criamos no passo 2.
- $z \sim Multinomial(\theta)$
- Em seguida sorteamos w usando $p(w|z, \beta)$, onde β é uma matriz $k \times m$ onde β_{ij} é a probabilidade de a palavra j ser selecionada dentro do tópico i (m é a quantidade de palavras no corpus).

Latent Dirichlet Allocation - LDA

- P/ estimar o modelo precisamos encontrar o α e β que maximizam a probabilidade de observar os ws:
- $p(\boldsymbol{w}|\alpha, \beta) = \frac{\Gamma(\Sigma_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \int \left(\prod_{i=1}^{k} \theta_i^{\alpha_i - 1} \right) \left(\prod_{n=1}^{N} \sum_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right)$
- (V é a quantidade de palavras únicas no vocabulário. Os demais termos foram definidos anteriormente.)

Latent Dirichlet Allocation - LDA

- P/ estimar o modelo precisamos encontrar o α e β que maximizam a probabilidade de observar os ws:
- $p(\boldsymbol{w}|\alpha,\beta) = \frac{\Gamma(\Sigma_i \alpha_i)}{\prod_i \Gamma(\alpha_i)} \int \left(\prod_{i=1}^{k} \theta_i^{\alpha_i - 1} \right) \left(\prod_{n=1}^{N} \sum_{i=1}^{k} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right)$
- (V é a quantidade de palavras únicas no vocabulário. Os demais termos foram definidos anteriormente.)
- Essa equação é intratável: não podemos achar uma resposta analítica.
- Solução: tentativa e erro. Várias possibilidades. Ex.: Hoffman, Blei, and Blach (2010).

Latent Dirichlet Allocation - LDA

- O output é:
- ... uma matriz $m \times k$ de palavras X tópicos, onde cada célula é a probabilidade de a palavra i ser sorteada se nós sortearmos uma palavra do tópico i; e
- ... uma matriz $k \times n$ de tópicos X documentos, onde cada célula é a proporção de palavras do tópico i no documento j.
- Ou seja, diferente de LSA aqui o output tem uma interpretação natural.