CSC411: Final Review

James Lucas & David Madras

December 3, 2018

Agenda

1. A brief overview
2. Some sample questions

Basic ML Terminology

The final exam will be on the entire course; however, it will be more heavily weighted towards post-midterm material. For pre-midterm material, refer to the midterm review slides on the course website.

- Feed-forward Neural Network (NN)
- Activation Function
- Backpropagation
- Fully-connected vs. convolutional NN
- Dimensionality Reduction
- Principal Component Analysis (PCA)
- Autoencoder
- Generative vs. Discriminative Classifiers
- Naive Bayes
- Bayesian parameter estimation
- Prior/posterior distributions
- Gaussian Discriminant Analysis (GDA)

Basic ML Terminology

The final exam will be on the entire course; however, it will be more heavily weighted towards post-midterm material. For pre-midterm material, refer to the midterm review slides on the course website.

- K-Means (hard and soft)
- Latent variable/factor models
- Clustering
- Gaussian Mixture Model (GMM)
- Expectation-Maximization (EM) algorithm
- Jensen's Inequality
- Matrix factorization
- Matrix completion
- Gaussian Processes
- Kernel trick
- Reinforcement learning
- States/actions/rewards
- Exploration/exploitation

Some Questions

Question 1

True or False:

1. PCA always uses an invertible linear map
2. K-Means will always find the global minimum
3. Naive Bayes assumes that all features are independent

Some Questions

Question 1

True or False:

1. PCA always uses an invertible linear map *False*
2. K-Means will always find the global minimum *False*
3. Naive Bayes assumes that all features are independent *False*

Some Questions

Question 1

True or False:

1. PCA always uses an invertible linear map *False*
2. K-Means will always find the global minimum *False*
3. Naive Bayes assumes that all features are independent *False*

Question 2

1. How can a generative model $p(\mathbf{x}|y)$ be used as a classifier?

Some Questions

Question 1

True or False:

1. PCA always uses an invertible linear map *False*
2. K-Means will always find the global minimum *False*
3. Naive Bayes assumes that all features are independent *False*

Question 2

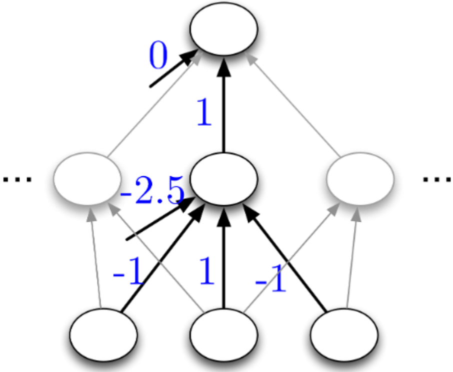
1. How can a generative model $p(\mathbf{x}|y)$ be used as a classifier?
2. Give one advantage of Bayesian linear regression over ML linear regression. Give a disadvantage.

Subject Areas

1. Neural Networks
2. PCA
3. Probabalistic Models
4. Latent Variable Models
5. Bayesian Learning
6. Reinforcement Learning

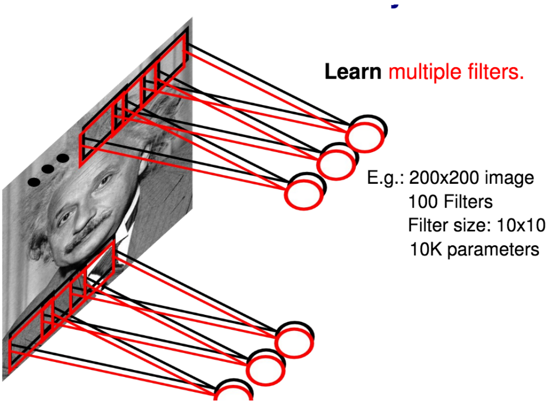
Neural Networks

1. Weights and neurons
2. Activation functions
3. Depth and expressive power
4. Backpropagation



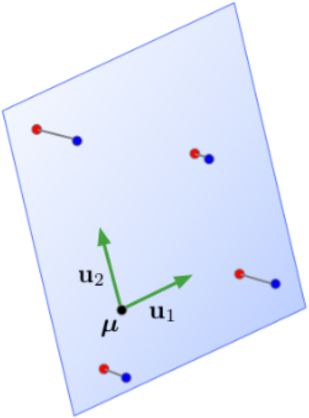
Convolutional Neural Networks

1. Convolutional neural network (CNN) architecture
2. Local connections/convolutions/pooling
3. Feature learning

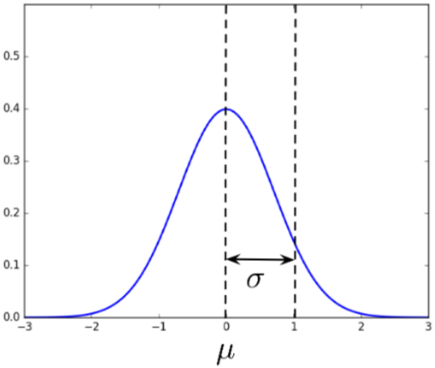


Principal Component Analysis (PCA)

1. Dimensionality reduction
2. Linear subspaces
3. Spectral decomposition
4. Autoencoders

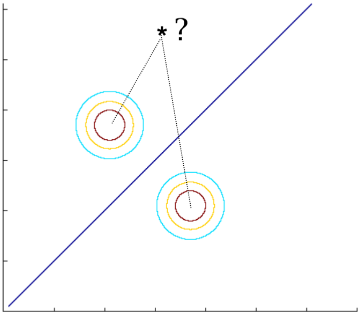


Probabilistic Models



1. Maximium Likelihood Estimation (MLE)
2. Generative vs. discriminative classification

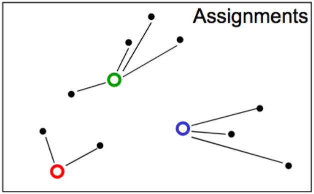
Probabilistic Models (continued)

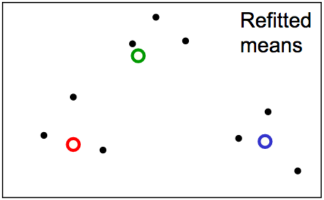


1. Bayesian parameter estimation
2. Choosing priors
3. Maximium A Posteriori (MAP) Estimation
4. Gaussian Discriminant Analysis
5. Decision Boundary

K-Means

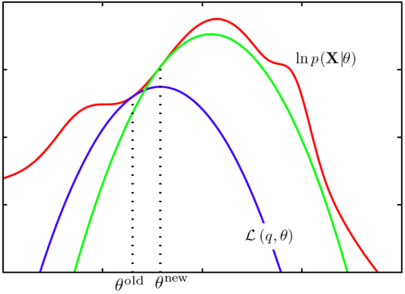
1. Latent-variable models for clustering
2. Initialization, assigment, refitting
3. Convergence
4. Soft vs. hard K-means



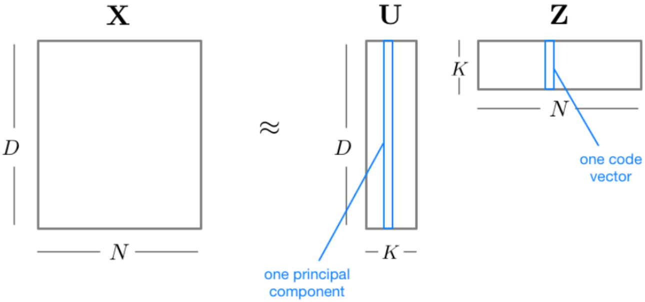


Expectation-Maximization (EM)

1. Gaussian Mixture Model (GMM)
2. E-Step, M-Step
3. GMM vs. K-Means
4. Autoencoders



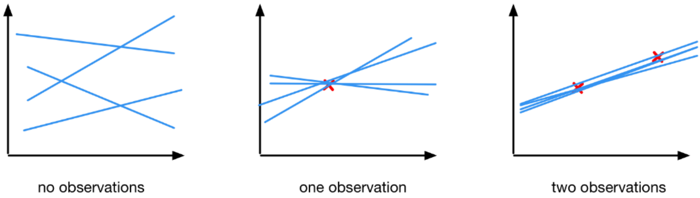
Matrix Factorization



1. Rank-k approximation
2. Matrix completion (movie recommendations)
3. Latent factor models
4. Alternating Least Squares (EM)
5. K-Means, Sparse Coding

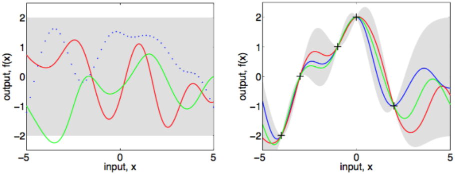
Bayesian Linear Regression

1. Posterior distribution over the parameters
2. Bayesian decision theory
3. Bayesian optimization



Gaussian Processes

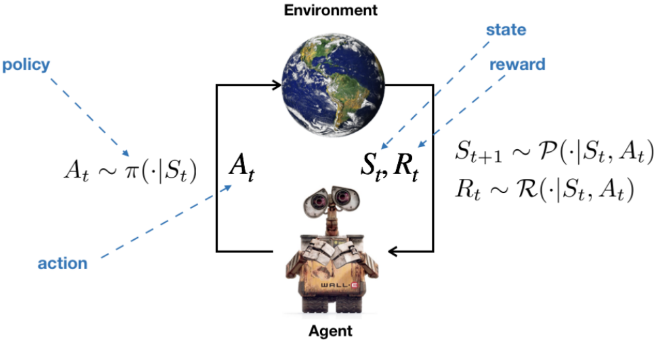
1. Distribution over functions!
2. Every point has a Gaussian distribution
3. Kernel functions



(a), prior (b), posterior

Reinforcement Learning

1. Choosing actions to maximize long-term reward
2. States, actions, rewards, policies
3. Value function and value iteration
4. Batch vs. Online RL
5. Exploration vs. Exploitation



Sample Question 1

Consider a 2-layer neural network, f, with 10-100-100 units in each layer respectively. We denote the weights of the network as $W^{(1)}$ and $W^{(2)}$.

a) What are the dimensions of $W^{(1)}$ and $W^{(2)}$? How many trainable parameters are in the neural network (ignoring biases)?

We will now replace the weights of f with a simple *Hypernetwork*. The Hypernetwork, h, will be a two layer network with 10 input units, 10 hidden units, and K output units where K is equal to the total number of trainable parameters in f. In each forward pass, the output of h will be reshaped and used as the weights of f.

b) How many parameters does h have (ignoring biases)?

c) How might we change the output layer to reduce the number of parameters? State how many trainable parameters h has with your suggested method. (HINT: use matrix factorization)

Q1 Solution

a) $W^{(1)} \in \mathbb{R}^{10\times 100}$, $W^{(1)} \in \mathbb{R}^{100\times 100}$. Total parameters: $10 \times 100 + 100 \times 100 = 11000$.

b) Total parameters: $10 \times 10 + 10 \times 11000 = 110100$.

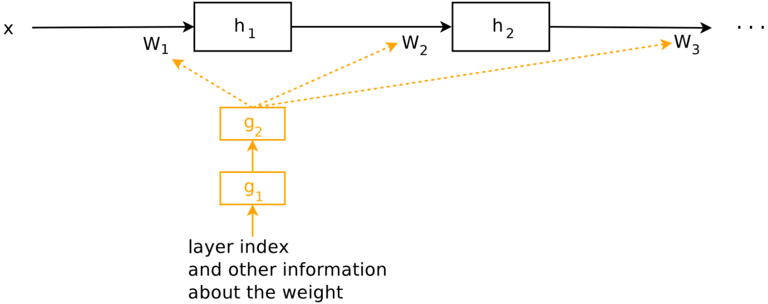
c) Output low rank approximations to each weight matrix. Instead of outputting $W^{(l)}$, output $U^{(l)}$ and $V^{(l)}$ such that $W^{(l)} \approx U^{(l)}V^{(l)}$. For example:

$$U^{(1)} \in \mathbb{R}^{10\times 2} \quad V^{(1)} \in \mathbb{R}^{2\times 100} \quad U^{(2)} \in \mathbb{R}^{100\times 2} \quad V^{(2)} \in \mathbb{R}^{2\times 100}$$

Now the total number of parameters is:
$10 \times 10 + 10 \times (2 \times 10 + 2 \times 100 + 100 \times 2 + 2 \times 100) = 6300$

Quick interlude: Hypernetworks

This isn't quite how Hypernetworks typically work...



See Ha et al. 2016 for details

Sample Question 2

a) State what conditions a function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ must satisfy to be a valid kernel function.

b) Prove that a symmetric matrix $K \in \mathbb{R}^{d \times d}$ is positive semidefinite if and only if for all vectors $\mathbf{c} \in \mathbb{R}^d$ we have $\mathbf{c}^T K \mathbf{c} \geq 0$.

Q2 Solution

a) Its Gram matrix, given by $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ must be positive semidefinite for any choices of $\mathbf{x}_1, \ldots, \mathbf{x}_d$.

b) First $\Rightarrow$: If K is PSD then there exists an orthonormal basis of eigenvectors $\mathbf{v}_1, \ldots, \mathbf{v}_d$ with non-negative eigenvalues $\lambda_1, \ldots, \lambda_d$. We can write any vector $\mathbf{c}$ in this basis: $\mathbf{c} = \sum_{i=1}^{d} a_i \mathbf{v}_i$. Then,

$$\mathbf{c}^T K \mathbf{c} = (\sum_{i=1}^{d} a_i \mathbf{v}_i)^T K (\sum_{i=1}^{d} a_i \mathbf{v}_i) = \sum_{i=1}^{d} a_i a_j \mathbf{v}_i^T K \mathbf{v}_j = \sum_{i=1}^{d} a_i a_j \mathbf{v}_i^T \lambda_j \mathbf{v}_j$$

As each of the $\mathbf{v}$'s are orthonormal, this sum is equal to $\sum_{i=1}^{d} a_i^2 \lambda_i \geq 0$.
For $\Leftarrow$: Pick $\mathbf{c} = \mathbf{v}$, some eigenvector. Then $\mathbf{v} K \mathbf{v} = \lambda \mathbf{v}^T \mathbf{v} \geq 0 \Rightarrow \lambda \geq 0$.