Machine Learning
Midterm Practice Problems

Some of these sample problems had been used in past exams and are provided for practice, in addition to the homework problems which you should also review. A typical exam would have around 5 questions worth a total of 100 points. The exam is closed book, no electronics. A single 8“x11” sheet of paper with typed or handwritten notes on both sides is allowed.

Contents

1. Math and Probability Basics

Q1.1 Definitions

[a] Give the definition of an orthogonal matrix.

[b] Give the definition of an eigenvector and eigenvalue.

[c] How is the probability density function different from the cumulative probability distribution?

[d] What is a 'singular' matrix?

[e] Give the definition of Baye's Rule.

Answer: see Bishop or Wikipedia for these definitions. These and other pre-requisites you should brush up on were discussed in the first week of lectures.

Q1.2 Covariance Matrix

Recall that in the derivation of normal equations in class, we used the fact that the data covariance matrix, i.e. a matrix whose element in the *k, j* position is the covariance between the k-th and j-th elements of the random input vector, is given b

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

where $x^{(i)}$ is the ith $nx1$ input vector, m is the number of input vectors in the dataset, and X is the mxn design matrix. Show that the above equality is true. Show all your steps.

Answer: using linear algebra rules, we can prove this by re-writing the vector multiplication inside the sum as a matrix, then re-writing the sum of matrices as a single matrix with sums as its elements, and finally as a product of two matrices.

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

$$= \sum_{i=1}^{m} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}^{(i)} \begin{bmatrix} x_1 & \dots & x_n \end{bmatrix}^{(i)}$$

$$= \sum_{i=1}^{m} \begin{bmatrix} x_1 x_1 & \dots & x_1 x_n \\ \vdots & \ddots & \vdots \\ x_n x_1 & \dots & x_n x_n \end{bmatrix}^{(i)}$$

$$= \begin{bmatrix} \sum_i {x_1}^{(i)} {x_1}^{(i)} & \dots & \sum_i {x_1}^{(i)} {x_n}^{(i)} \\ \vdots & \ddots & \vdots \\ \sum_i {x_n}^{(i)} {x_1}^{(i)} & \dots & \sum_i {x_n}^{(i)} {x_n}^{(i)} \end{bmatrix}$$

$$= \begin{bmatrix} | & & | \\ x^{(i)} & \dots & x^{(m)} \\ | & & | \end{bmatrix} \begin{bmatrix} — & x^{(i)} & — \\ & \vdots & \\ — & x^{(m)} & — \end{bmatrix}$$

$$= X^T X$$

Q1.3 Matrix Norm

The trace of a square matrix $A \in \mathbb{R}^{nxn}$ is defined as the sum of diagonal entries, or

$$\mathrm{tr}A = \sum_{i=1}^{n} A_{ii}.$$

Prove the following fact

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2} = \sqrt{\mathrm{tr}(A^T A)}.$$

where $\|A\|_F$ is the matrix Frobenius norm. Show all your steps.

Answer: this is straightforward to show using linear algebra rules. Without providing all the details here, the idea is to first write down each diagonal element of the matrix $C = A^T A$ as $C_{kk} = \sum_{i=1}^{n} A_{ik} A_{ik}$ and then compute the trace as the sum of these diagonal elements, taking all sums outside.

Q1.4 Flu Virus Test

After your yearly checkup, the doctor has bad news and good news. The bad news is that you tested positive for a flu virus, and that the test is 99% accurate (i.e., the probability of testing positive given that you have the virus is 0.99, as is the probability of testing negative given that you don't have the disease). The good news is that this is a rare virus, striking only one in 10,000 people (10^{-4}). What are the chances that you actually have the disease? Show your calculations as well as giving the final result. *Hint: use Baye's Rule.*

Answer: This is a straightforward application of the rules of probability. Let T be the binary variable with value 1 if your test is positive and 0 if negative, and D be a variable with value 1 if you have the disease and 0 of you don't. We are given

p(T=1|D=1)=0.99, p(T=0|D=1)=0.01
p(T=0|D=0)=0.99, p(T=1|D=0)=0.01
p(D=1)=0.0001, p(D=0)=0.9999

We want the probability of having the disease given the test was positive, or p(D=1|T=1), which is given by

$$p(D=1|T=1) = \frac{p(T=1, D=1)}{p(T=1)} = \frac{p(T=1|D=1)p(D=1)}{p(T=1)}$$
$$p(T=1) = p(T=1|D=1)p(D=1) + p(T=1|D=0)p(D=0)$$
$$p(T=1) = 0.99 * 0.0001 + 0.01 * 0.9999 = 0.000099 + 0.009999$$
$$p(D=1|T=1) = \frac{0.000099}{0.000099 + 0.009999} \approx 0.0098$$

An answer that left the fraction un-simplified is given full points.

2. Gradient Descent

Q2.1 Gradient Descent for a Cost Function

Suppose we have a cost function

$$J(\theta) = \frac{1}{m}(\sum_{i=1}^{m} x_i^T \theta + by_i) + \frac{1}{2}\theta^T A\theta,$$

where $\theta \in \mathbb{R}^n$ is the parameter vector, $x_i \in \mathbb{R}^n$, $y_i \in \mathbb{R}$, $\{x_i, y_i\}$ are m training data points, $A \in \mathbb{R}^{nxn}$ is a symmetric matrix, and $b \in \mathbb{R}$. We want to find parameters θ using gradient descent.

a) [3 points] Give the pseudo code for the gradient decent algorithm for a **generic** cost function $J(\theta)$ (not the specific one above).

Answer: initialize θ, then update $\theta := \theta - \alpha \frac{\partial}{\partial \theta} J(\theta)$ until convergence.

b) [3 points] For the specific function above, what is the vector of partial gradients of the cost function, i.e. the vector with the jth element equal to $\frac{\partial}{\partial \theta_j} J(\theta)$?

Answer: $\frac{\partial}{\partial \theta} J(\theta) = \frac{1}{m}\sum_i x_i + \frac{1}{2} 2A\theta = \frac{1}{m}\sum_i x_i + A\theta$

c) [3 points] What is the design matrix? Describe its entries and give its dimensions.

Answer: matrix $X \in \mathbb{R}^{mxn}$ with the ith row equal to x_i^T.

d) [3 points] Re-write the expression for the gradient without using the summation notation $\sum$. *Hint: use the design matrix X.*

Answer: $A\theta + \frac{1}{m}\sum_i x_i = A\theta + \frac{1}{m} X^T \mathbf{1}$, where $\mathbf{1}$ is an m-dimensional vector with all entries equal to 1. As a sanity check, the result is a nx1 vector, as it should be.

e) [3 points] Suppose we run gradient descent for two iterations. Give the expression for θ after two updates, with step size $\alpha = 1$ and initial value of $\theta = \mathbf{0}$ (vector of zeros).

Answer: $\theta \coloneqq \theta - \alpha(A\theta + \bar{x})$; where $\bar{x} = \frac{1}{m} X^T \mathbf{1}$
$\theta^{(1)} \coloneqq 0 - 1(A0 + \bar{x}) = -\bar{x}$,
$\theta^{(2)} \coloneqq -\bar{x} - 1(-A\bar{x} + \bar{x}) = -\bar{x} + A\bar{x} - \bar{x} = (A - 2)\bar{x}$

f) [3 points] How do we know when the algorithm has converged?

Answer: when the cost or parameter is not changing much between iterations.

g) [3 points] Give the closed-form solution for θ. You do not need to prove it is the minimum of the cost.

Answer: Setting the gradient to zero, $A\theta + \bar{x} = 0$ and solving, we get
$\theta = -A^{-1}\bar{x}$

3. Regression and Classification

Q3.1 Linear Regression: Online Art Auction

Imagine you work for an online art auctioneer. You would like to estimate the price y that a piece of art will sell for in an auction (in dollars), based on the following features:

x_1= type of art (out of 15 types such as 1:painting, 2:sculpture, etc.),
x_2= artist popularity (rank out of 100 artists),
x_3= estimated value in dollars,
x_4= days in the auction,
x_5= previously owned (binary),
x_6= is abstract (binary),
...etc.

For example, a feature vector for the i^{th} item could be $x^{(i)} = [1, 28, 1700, 5, 1, 0, \dots]$. You have collected data points from previous auction sales, $(x^{(i)}, y^{(i)})$, $i = 1, \dots, m$.

a. [3 points] You decide to use a linear regression model, $y = \sum_{j=0}^{n} \theta_j x_j$. In what circumstances should you use gradient descent vs normal equations to fit the parameters?

Answer: Use gradient descent when the number of features n is large, i.e. the matrix X^TX is too large to invert ($O(n^3)$), otherwise use normal equations.

b. [3 points] Suppose you decide to use gradient descent. How can you tell if it is converging?

Answer: By plotting the cost as a function of the number of iterations: convergence is likely when the decrease in cost diminishes.

c. [3 points] Suppose you're monitoring convergence and find it is slow. Name two things you can try to speed it up (be specific).

Answer: 1) Try a larger value for the step size; 2) In this case, some of the features have larger scale which could cause slow convergence, so we could use feature re-scaling to normalize all features, e.g. to [-1 1]

d. [3 points] You want to add new features to improve your predictor. You consider adding total minutes spent in the auction. Is this a good idea? Why or why not?

Answer: No, because it is redundant with (a constant multiple of) the time in months and would not add new information.

e. [3 points] Your boss does not know how much to trust your prediction $y^{test} = \$15{,}000$ for a certain watercolor painting. She asks you to estimate the probability of the painting selling for more than $\$20{,}000$. Give the equation for this probability, using a linear regression model that assumes the outputs have Gaussian noise with variance β^{-1}.

Answer: Assuming that the outputs y contain Gaussian noise gives us a Normal predictive distribution,
$p(y^{test}|x^{test}, \theta, \beta^{-1}) = N(y^{test}|h_\theta(x^{test}), \beta^{-1})$.
Therefore the probability of y^{test} being more than 20,000 is
$p(y^{test} > 20000) = \int_{20000}^{\infty} N(y^{test}|15000, \beta^{-1})dy$.

f. [2 points] What is the probability the painting will sell for more than $15,000?

Answer: $p(y^{test} > 15000) = \int_{15000}^{\infty} N(y^{test}|15000, \beta^{-1})dy$=0.5

g. [3 points] Suppose you now want to estimate the probability that a piece of art does not get any bids, $p(no\_bids|x)$, based on historic data. What sort of features and machine learning method should you use?

Answer: The same features could be used as in the previous example; any classification algorithm could be applied, e.g. logistic regression or LDA or neural net, etc., with a binary class label (got bids or didn't get bids).

Q3.2 Softmax Classifier

Typically, we use the *softmax* function to model the probability of one of several classes given the input. Instead, consider what would happen if a binary classifier with output labels $j \in \{0,1\}$ used the softmax function to model the probability of the binary label $y = j$ given the input:

$$P(y = j|x) = \frac{e^{{w_j}^T x}}{\sum_{j=0,1} e^{{w_j}^T x}}$$

Where w_j is the parameter vector of the jth class.

a) [5 points] Show that the solution to this problem can be expressed as a logistic regression classifier with parameter w, and give the expression for w.

Answer: we can use a logistic regression classifier with parameter $w = w_1 - w_0$

$$\frac{e^{{w_1}^T x}}{\sum_{j=0,1} e^{{w_j}^T x}} = \frac{1 * (e^{{w_1}^T x})}{(1 + e^{(w_0 - w_1)^T x}) * (e^{{w_1}^T x})} = \frac{1}{(1 + e^{(w_0 - w_1)^T x})} = \frac{1}{(1 + e^{-w^T x})}$$

Where we used the fact that $e^{({w_0}^T x - {w_1}^T x)} = e^{(w_0 - w_1)^T x}$

b) [5 points] Show that the posterior $P(y = j|x)$ is invariant if a constant vector b is added to both weight vectors w_j.

Answer: this is trivial to show using (a), since for any b, $w = w_0 + \mathrm{b} - w_1 - b = w_0 - w_1$

c) [5 points] (b) implies that the solution w_j is not unique. We can guarantee a unique solution by making the objective function regularized, e.g. using the squared norm regularizer. Write down the objective function and say whether you should minimize or maximize it.

Answer: Again, several possibilities, e.g. minimize

$$J = -\sum_i log P(y = y_i|x_i) + \lambda \|w\|^2$$

4. Overfitting and Regularization

Q4.1 Bias-Variance and λ

Alice has a binary classification dataset of m points with n-dimensional inputs.

a) [3 points] She has trained several regularized logistic regression models using regularization parameters $\lambda = 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}$. She computed the cross-validation (CV) and training errors for each value of λ, shown in the table below, but the rows are out of order. Fill in the correct values of λ for each row.

Train error	*CV error*	λ
80%	85%	
40%	45%	
70%	76%	
35%	50%	

Answer: $10^{-1}, 10^{-3}, 10^{-2}, 10^{-4}$

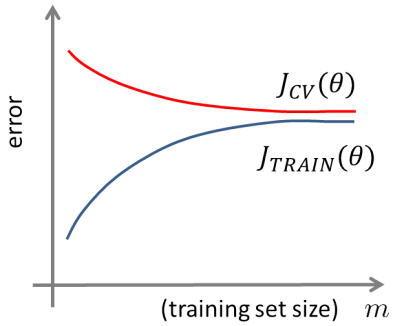
b) [3 points] Based on these results, which λ should she choose, and why?

Answer: 10^{-3}, the one with lowest CV error has best generalization

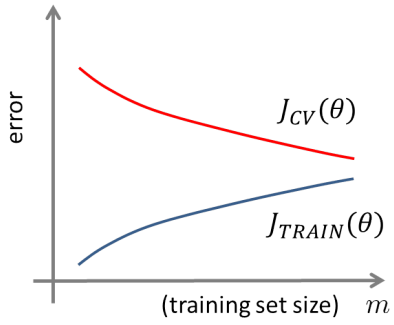
c) [3 points] Which of the four models will have the highest error due to variance? Why?

Answer: 10^{-4}, it is has the least regularization and is the most complex

d) [3 points] Alice also plotted learning curves for the models with $\lambda = 10^{-1}, 10^{-4}$. Match each plot with the correct value, and explain why it matches.



$\lambda =$



$\lambda =$

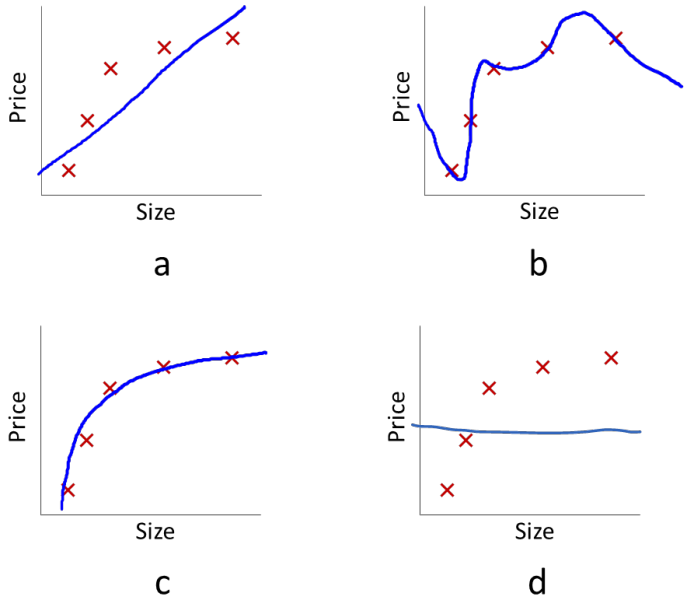
Answer: Left: 10^{-1} because it has high bias, more data doesn't help; Right: 10^{-4} because it has high variance, more data may help.

Q4.2 Regularization for Linear Regression

Alice is trying to fit a linear regression model to predict house price based on size using polynomial features. Since her training dataset is very small, she is applying regularization. She fit several models by minimizing the cost function

$$J(\theta) = \frac{1}{2m}\left[\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n}\theta_j^2\right]$$

for $\lambda = 10^0, 10^1, 10^2, 10^3$. The following are sketches of the resulting models.



a) [3 points] Which value of λ goes with each of the plots? (Write it next to the plot)

Answer: a: 10^2 b: 10^0 c: 10^1 d: 10^3

b) [3 points] Alice tries her model on a test set. Which model will have the highest error due to bias?

Answer: d

c) [3 points] Which model will have the highest error due to variance?

Answer: b

d) [3 points] Which model, if any, will always have zero test error?

Answer: none

5. Maximum Likelihood Principle

Q5.1 ML for Probabilistic Linear Regression

Recall that probabilistic linear regression defines the likelihood of observing outputs $t^{(i)} \in \mathbb{R}$ given inputs $x^{(i)} \in \mathbb{R}^p$, where $i = 1, \dots, m$ and m is the number of samples in the dataset, as

$$p(t_1, \dots, t_m | x_1 \dots, x_m, \theta, \beta) = \prod_{i=1}^{m} N(t^{(i)} | h(x^{(i)}),\ \beta^{-1})$$

where $h(x)$ is the linear regression hypothesis, θ, β are parameters and $N(x|\mu, \sigma^2)$ is the normal (Gaussian) probability density with mean μ and variance σ^2. Here $\beta = \sigma^{-2}$ is the inverse variance of the Gaussian noise that we assume is added to the data.

(a) [8 points] Find β_{ML}, the maximum likelihood solution for β. *Hint: maximize log likelihood with respect to only β.*

Answer: maximize log likelihood w.r.t. β. The likelihood function is:

$$\ln p(\boldsymbol{t}|\boldsymbol{x}, \theta, \beta) = -\frac{\beta}{2}\sum_{i=1}^{m}\left(h(x^{(i)}) - t^{(i)}\right)^2 + \frac{m}{2}\ln\beta - \frac{m}{2}\ln(2\pi)$$

Take partial derivative w.r.t. β and set it to 0, then solve for w.r.t. β

$$-\frac{1}{2}\sum_{i=1}^{m}\left(h(x^{(i)}) - t^{(i)}\right)^2 + \frac{m}{2}\frac{1}{\beta} = 0$$

$$\frac{1}{\beta_{ML}} = \frac{1}{m}\sum_{i=1}^{m}\left(h(x^{(i)}) - t^{(i)}\right)^2$$

To get β_{ML} we just need to take the inverse of the right-hand side.

(b) [2 points] What is the interpretation of the solution β_{ML}? Explain in one sentence.

Answer: It is the inverse of the average squared error of the prediction.

Q5.2 ML for Linear Regression with Multivariate Outputs

Consider a probabilistic linear regression model for a multivariate p-dimensional target variable $t = [t_1, \dots, t_p]^T$ that has a Gaussian distribution over t of the form

$$p(t|W, \Sigma) = N(t|y(\phi(x), W), \Sigma)$$

where $\phi(x)$ is a basis function representation of the input, and

$$y(x, W) = W^T \phi(x)$$

W is a matrix of parameters and Σ is the covariance parameter matrix. We are given a training dataset of input basis vectors x_n and corresponding target vectors t_n, $n = 1, \dots, N$. Show that the Maximum Likelihood solution W_{ML} for the parameter matrix W has the property that each column is given by

$$w_{ML} = (\Phi^T \Phi)^{-1} \Phi^T t$$

where Φ is the design matrix. You do not need to provide a solution for Σ. Show all your steps..

Hint: the p-dimensional **multivariate normal distribution** is given by

$$N(x; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{p}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

Hint: you may also find some of the matrix differentiation rules in the appendix helpful.

Answer: First we write down the Likelihood function, or the probability of the output datapoints given the input datapoints and parameters. The probability of a single datapoint is the multivariate normal

$$p(t|x; W, \Sigma) = \frac{1}{(2\pi)^{\frac{p}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}\left(t - W^T \phi(x)\right)^T \Sigma^{-1} \left(t - W^T \phi(x)\right)\right)$$

Then the likelihood is

$$L(t_1, \dots, t_N|x; W, \Sigma) = \frac{1}{(2\pi)^{\frac{p}{2}} |\Sigma|^{\frac{1}{2}}} \prod_{n=1}^{N} \exp\left(-\frac{1}{2}\left(t_n - W^T \phi(x_n)\right)^T \Sigma^{-1} (t_n - W^T \phi(x_n))\right)$$

and the log likelihood is

$$lnL(t_1, \dots, t_N|x; W, \Sigma)$$
$$= -\frac{Np}{2} ln(2\pi) \quad -\frac{N}{2} ln|\Sigma| \quad -\frac{1}{2} \sum_{n=1}^{N} \left(t_n - W^T \phi(x_n)\right)^T \Sigma^{-1} (t_n - W^T \phi(x_n))$$

Taking the derivative with respect to W, the first two terms vanish, and we set the rest to zero

$$\frac{\delta}{\delta W} lnL = \frac{\delta}{\delta W} \left(-\frac{1}{2} \sum_{n=1}^{N} \left(t_n - W^T \phi(x_n)\right)^T \Sigma^{-1} (t_n - W^T \phi(x_n))\right) = 0$$

Applying chain rule with $u = t_n - W^T \phi(x_n)$, and using the matrix derivative rules

$$-\frac{1}{2}\sum_{n=1}^{N}\frac{\delta}{\delta W}(t_n - W^T\phi(x_n))\frac{\delta}{\delta u}(u^T\Sigma^{-1}u) = 0$$

$$-\frac{1}{2}\sum_{n=1}^{N}(-\phi(x_n))(2u^T\Sigma^{-1}) = 0$$

$$-\frac{1}{2}\sum_{n=1}^{N}(-\phi(x_n))(2(t_n - W^T\phi(x_n))^T\Sigma^{-1}) = 0$$

Multiplying both sides by Σ^{-1}

$$\sum_{n=1}^{N}\phi(x_n)(t_n - W^T\phi(x_n))^T = 0$$

Let Φ be the Nxm data design matrix, where m is the dimension of $\phi(x_n)$, and T be the Nxp output matrix of output vectors t_n, then we can rewrite the above as

$$\Phi^T\Phi W = \Phi^T T$$

Then the ML solution is given by the matrix below, with each column equal to w_{ML} as required

$$W_{ML} = (\Phi^T\Phi)^{-1}\Phi^T T$$

Q5.3 ML for Poisson Regression

This problem asks you to derive the maximum likelihood solution for a Poisson hypothesis.

a) [3 points] Given a training set $\{x_i, y_i\}$, Poisson regression models the probability of observing an output given an input as $p(y_i|\lambda) = \frac{1}{y_i!}\lambda^{y_i}e^{-\lambda}$ where $\lambda = e^{\theta^T x_i}$ for some parameter vector θ. Derive the cost function $J(\theta)$ corresponding to maximizing the log likelihood for a **single training example**.

Answer: $\log(p(y_i|x_i,\theta) = \log\left(e^{y_i\theta^T x_i}\right) + \log\left(e^{-e^{\theta^T x_i}}\right) - \log(y_i!)$

$$= y_i\theta^T x_i - e^{\theta^T x_i} - \log(y_i!)$$

b) [3 points] The cost function in (b) has no closed form solution, so we must use an iterative method. Show the stochastic gradient descent update for this cost function.

Answer: $\frac{\partial}{\partial\theta}\log(p(y_i|x_i,\theta) = y_i x_i - x_i e^{\theta^T x_i} = \left(y_i - e^{\theta^T x_i}\right)x_i$, so the update is

$$\theta := \theta + \alpha\left(y_i - e^{\theta^T x_i}\right)x_i$$

Note that we are maximizing the function so we use gradient ascent, not descent.

6. Unsupervised Learning

Q6.1 Principle Component Analysis

a) PCA assumes a specific relationship between the unobserved latent coordinates z and the observed data points x. Express this relationship as an equation. Clearly identify and name the parameters which are learned.

Answer: $z_k = u^{(k)^T} \bar{x}$ where $\bar{x}$ is a normalized data point and $u^{(k)}, k = 1, \dots, K$ are the principle component vectors, which are the parameters that need to be learned.

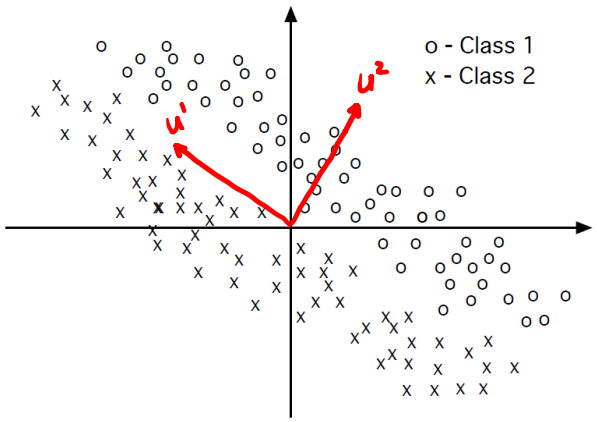
b) Name one objective function which could be minimized to learn the parameters of PCA.

Answer: Reconstruction error. Another objective is maximizing the total variance of projected points.

c) For a dataset of arbitrary points $x^{(1)}, \dots, x^{(m)}$, specify the steps of the PCA algorithm.

Answer: First, normalize the points to have zero mean and unit standard deviation in each coordinate. Then, compute the covariance matrix of the data, and decompose it with Singular Value Decomposition to obtain its eigenvectors/values. Finally, project each point onto the top k eigenvectors to obtain the lower-dimensional points. The eigenvalues can be used to determine how many components to keep in order to preserve a certain percentage of the total variance.

d) Suppose you are given 2D feature vectors for a classification task which are distributed according to the figure below. You apply PCA to the entire dataset. On the figure, draw all the PCA components.



e) In (d) above, could you use PCA components directly to classify the data (without training a classifier)? Explain.

Answer: yes, we can project the points onto the second eigenvector, u^2, and then threshold the one-dimensional points at 0 to assign the class label. Specifically, the classifier will assign labels $sign((u^2)^T x)$ where +1 corresponds to class 1.

Q6.2 Gaussian Mixture Models

a) Describe in words the two main steps of the Expectation Maximization algorithm used to solve Gaussian Mixture Models.

Answer: the E step estimates the values of the latent variables, the M step maximizes the likelihood of the data given the latent variables computed in the E step.

b) True or False: In the case of fully observed data, i.e. when all latent variables are observed, EM reduces to Maximum Likelihood.

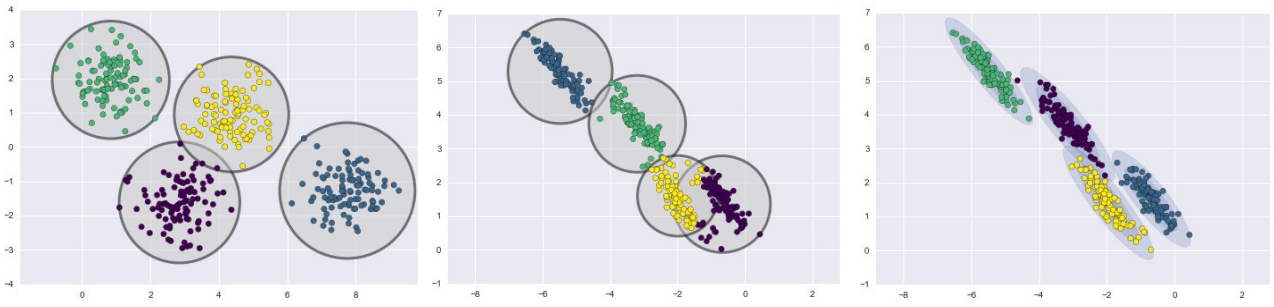
Answer: True

c) True or False: Since the EM algorithm guarantees that the value of its objective function will increase after each iteration, it is guaranteed to eventually reach the global maximum.

Answer: False. The algorithm is guaranteed to converge (as opposed to iterating forever) because the objective function is increased at each step, but it may converge to a local maximum rather than global one.

d) Sketch a dataset on which K-Means would work poorly but a Gaussian Mixture Model with the same number of clusters would do well. Describe why K-Means wouldn't work well.

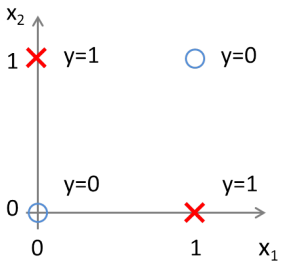
Answer: One advantage of GMM over k-means is that it accounts for data variance (or covariance in multiple dimensions.) One way to think about the k-means model is that it places a circle (or, in higher dimensions, a hyper-sphere) at the center of each cluster, with a radius defined by the most distant point in the cluster, as shown on the left. This works fine when your data is circular. However, when your data takes on different shape, you end up with something like the middle plot. On the other hand, a GMM can handle even non-circular clusters (right) by using the *covariance* to calculate the distance.

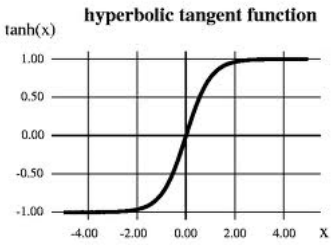


7. Neural Networks

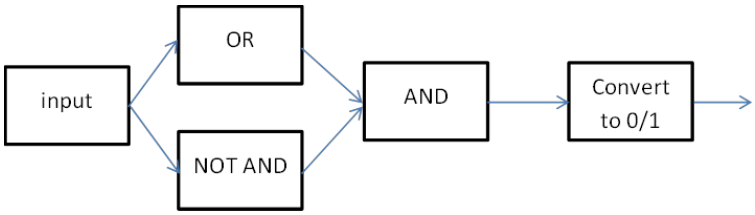
Q7.1 Neural Network for XOR

Design a neural network to solve the XOR problem, i.e. the network should output 1 if only one of the two binary input variables is 1, and 0 otherwise (see left figure). Use the hyperbolic tangent, or *tanh,* activation function in all nodes (right figure), which ranges in [-1, +1].



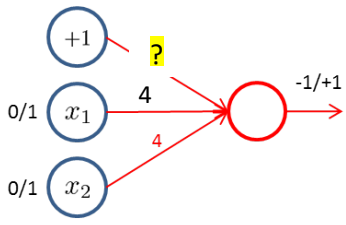


Note that (A XOR B) can be expressed as (A OR B) AND NOT(A AND B)), as illustrated below:

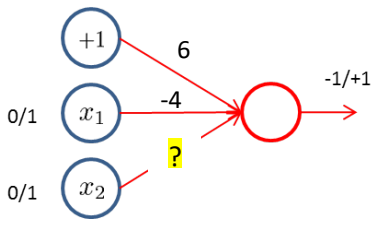


In the diagrams below, we filled in most of the tanh units' parameters. Fill in the remaining parameters, keeping in mind that tanh outputs +1/-1, not 0/1. Note that we need to appropriately change the second layer (the AND node) to take +1/-1 as inputs. Also, we must add an extra last layer to convert the final output from +1/-1 to 0/1. *Hint: assume tanh outputs* -1 *for any input* $x \leq -2$, $+1$ *for any input* $x \geq +2$, 0 *for* $x = 0$.

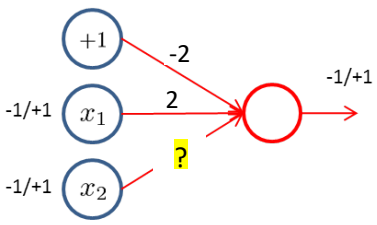
x_1 OR x_2



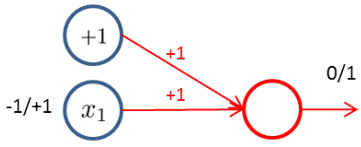
NOT(x_1 AND x_2)



x_1 AND x_2



convert to 0/1



Answer: -2, -4, +2

Q7.2 Computation Graph and Backpropagation

In class, we learned how to take a complex function that consists of multiple nested functions and represent it with a computation graph, which allows us to write down the forward and backward pass used to compute the function gradient.

a) Practice converting different functions $f_\theta(x) = f_k(f_{k-1}(\dots f_1(x)))$ of input vector x parametrized by θ to their computation graphs.

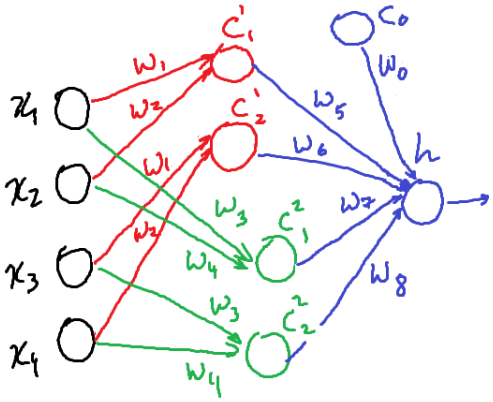
Answer: see lecture notes for examples.

b) For the computation graphs obtained in (a), write down the forward pass and the backward pass equations.

Answer: see lecture notes for examples.

Q7.3 Neural Network Architectures

a) Draw a convolutional network with input $x \in R^4$, one hidden layer with 2x1 filters and 2 channels with stride 2, and a fully-connected output layer with one neuron. How many parameters does this network have?



Answer: 9 parameters, see w's in the plot. The c's are the hidden convolutional units, with each channel sharing parameters. Note that the last layer should also have a "dummy" unit set to 1 and thus an extra parameter.

b) What algorithm is used for learning the parameters of a recurrent network? Name the algorithm and sketch out its main steps.

Answer: Backpropagation in time. The steps for computing the gradient for 1 training sequence are as follows. First, we unroll the network by replicating it once for each time step in our training sequence. Then we use regular backprop on the unrolled network to find the gradient. See lecture notes for more detail.

Appendix: Useful Formulas

Matrix Derivatives

For vectors x, y and matrix A,

$y = Ax$, then $\frac{\partial y}{\partial x} = A$

If $z = x^T Ax$, then $\frac{\partial z}{\partial x} = x^T(A + A^T)$. For the special case of a symmetric matrix A, $\frac{\partial z}{\partial x} = 2x^T A$.

Chain Rule: if z is a function of y, which is a function of A, then $\frac{\partial z}{\partial A} = \frac{\partial y}{\partial A}\frac{\partial z}{\partial y}$ (note the order).

Single-Dimension Normal Distribution

$$N(x; \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Multivariate Normal Distribution

The p-dimensional multivariate normal distribution with mean μ and covariance matrix Σ is given by

$$N(x; \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$