CSE517a Final

Good luck!

May 2015

NAME:	
Student ID:	
Email:	

General ML	
Kernls	
Neural Networks	
GPs	
Unsupervised Learning	
TOTAL	

1 [??] General Machine Learning

Either circle **T** or **F**. Questions declared as **True** require no explanation (worth 1 point). Questions declared as **False** require a **one sentence** explanation (worth 2 points).

T/F The fundamental difference between Bayesian and Frequentist Statistics boils down to the definition of good pop music. (For example, "Bayesians" love Meghan Trainor's song "All about the Bayes" and just cannot hear enough of it, whereas "Frequentists" believe strongly that this song has been played way too frequently on the media in recent months. Frequentists are also concerned that certain passages may be inappropriate, in particular when the singer refers to *shaking her posterior*, which should be changed to *likelihood*.)
False the difference between Bayesian and Frequentist statistics boils down to if a random variable has to be associated with a random event or not.

T/F The difference between Parametric and Non-parametric algorithms is that non-parametric algorithms have no hyper-parameters to tune.
False, the difference is in the number of parameters. Parametric algorithms have a fixed number of parameters, independent of the training set size. In non-parametric algorithms the number of parameters grows with the training set size.

T/F Platt scaling is used to scale features to be between 0 and 1.
False, PS is used to scale the outputs of a classifier (*e.g.* SVM) to become probabilities.

T/F One advantage of 1 *vs.* 1 multi-class classification over 1 *vs. all* is that the problems are more likely to be class balanced.
True.

T/F After convergence, the K-means cluster centers are always original data points.
False, they are the average of all inputs in the cluster - very unlikely to fall exactly onto a data point.

T/F SVD decomposes a matrix $\mathbf{X}$ (with data as column vectors) into three terms $\mathbf{X} = \mathbf{USV}^\top$. The matrix $\mathbf{S}$ is diagonal and non-negative, the matrix $\mathbf{U}$ is the projection matrix of PCA the matrix V is the whitened data.
True

T/F The i^{th} largest eigenvalue of the covariance corresponds to the amount of variance captured in the i^{th} principal component.
True

T/F TheSupport Vector Machine with the RBF kernel is non-parametric.
True.

T/F K-means is a non-parametric algorithm
False it is parametric. The number of parameters is exactly one mean vector for each of the k clusters — independent of the input size.

T/F Kernel SVM (in the dual) with a linear kernel is always slower than training the linear SVM directly in the primal (without a kernel matrix), because the kernel matrix has size $n \times n$.
False for small training set sizes n with high dimensionality d the kernel version can be a lot faster than the li near version because you don't have to repeatedly compute the inner products $O(d)$ between the weight and input vectors.

T/F Any real *symmetric* matrix is a well defined kernel.
False it has to be positive semi-definite (*i.e.* all eigenvalues are non-negative).

T/F A sign for a high variance scenario is that the training error is higher than the testing error.
False a typical sign is that there is a large gap between training and testing error.

T/F An effective way to fight a **high bias** scenario is to substantially increase the amount of training data.
False in the high bias scenario the training error is typically already undesirably high. Adding more training data will only increase the training error, which is typically a lower bound on the testing error.

T/F Performing leave-one-out cross validation with the squared loss requires you to train n classifies. Each time you leave out one point, train the classifier on the remaining $n-1$ points and compute the error of the resulting classifier on the

left out point. The error is the average error across all n classifiers.
False For the squared loss the leave-one-out loss can be computed in closed form (HW3).

2 [20] Kernels

1. (5) Assume Ω is a set of all words in the English dictionary, $|\Omega| = d$. We define a text document as the set of all the words that are in the document, $S \subseteq \Omega$. Proof that the following kernel over such documents is well-defined (*i.e.* positive semi-definite)
$$k(S_1, S_2) = \exp\left(\frac{|S_1 \cap S_2|}{|S_1|\ |S_2|}\right).$$
(Hint: It helps to write $\Omega = \{\omega_1, \ldots, \omega_n\}$ and represent a set S as a binary vector $x \in \{0,1\}^d$, where the i^{th} dimension $x_i = 1$ if and only if $\omega_i \in S$.)

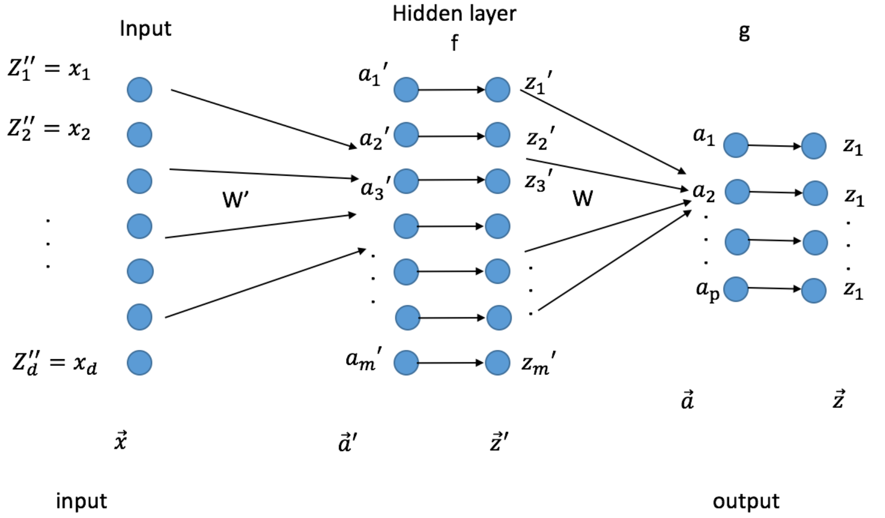
2. (10) Assume you are given a data set $D = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_n, y_n)\}$ with $\vec{x}_i \in \mathcal{R}^d$ and $y_i \in \{+1, -1\}$. The Perceptron algorithm learns a separating hyper-plane in the following way:
   (a) Initialize $\mathbf{w} = \vec{0}$
   (b) Pick $(\vec{x}_i, y_i)$ randomly from D.
   (c) If $y_i \mathbf{w}^\top \vec{x}_i \leq 0$ then make the update $\mathbf{w} \leftarrow \mathbf{w} + y_i \vec{x}_i$, otherwise do nothing.
   (d) Goto step (b) and repeat until convergence.

   Show that this algorithm can be kernelized. First show that $\mathbf{w}$ can be written as a linear combination of the inputs. State the kernelized classifier $h()$ and the kernelized version of the learning algorithm.

3. (5) Joe wants to combine the power of kernel machines with that of neural networks. He trains a neural network with one hidden layer on his training data. Then he defines the mapping $f(\vec{x}) = \vec{z}'$ to be the mapping of his input to the first hidden layer (after the transition function has been applied). (See the neural network figure on the next page for an illustration.) He then defines his kernel matrix $k(\vec{x}_1, \vec{x}_2) = f(\vec{x}_1)^\top f(\vec{x}_2)$ to train an SVM. Is this kernel function well defined?

   Yes, it is. After all it corresponds to inner products in a well defined space.

3 [18] Neural Networks



1. (3) Write z and z' in terms of x, a', a, W', W and the transition functions f and g. $z = g(Wf(W'x))$, $z' = f(W'x)$

(5) Given $\delta_j = \frac{\partial L}{\partial a_j}$, derive $\delta_j' = \frac{\partial L}{\partial a_j'}$.

$$\begin{aligned}\delta_j' = \frac{\partial L}{\partial a_j'} &= \sum_{i=1}^{m} \frac{\partial L}{\partial a_i} \frac{\partial a_i}{\partial z_j'} \frac{\partial z_j'}{\partial a_j'} \\ &= \sum_{i=1}^{m} \delta_i W_{ij} \frac{\partial z_j'}{\partial a_j'} \\ &= f'(a_j') \sum_{i=1}^{m} \delta_i W_{ij} \\ &= f'(a_j')(W^T\delta)_j\end{aligned}$$

2. (4) Batman is frustrated about how slow his super deep Neural Network is. He decides to remove all non-linear transition functions to speed it up. To his surprise, this change drastically increases his training error. Can you explain why? The bias of the neural network will increase dramatically. Without transition functions, the network collapses to a linear classifier.

3. (3) Rectified Linear Units have become more popular transition function than Sigmoids. Name one advantage of Rectified linear Units over sigmoid functions. Sigmoid functions cause weights to saturate. This is because the gradient becomes very small when the input is far from 0.

4. (3) Why is it often impractical to use Newton's method to train a neural network? Because there is such a large number of parameters, the Hessian matrix is too large to be computed practically.

4 [11] Gaussian Processes

1. (5) What are the assumptions of Gaussian Process Regression (with mean $\vec{\mu}_x$ and variance $\mathbf{K}_x$) about the distribution of the labels $y_1, \ldots, y_n$ and a test label y_t?
   It assumes that the data is distributed from a multi-variate Gaussian Distribution.

2. (5) The mean prediction of GPR is identical to kernel regression. So why can't we easily use kernel regression for hyper-parameter optimization with the Upper Confidence Bound (UCB) exploration / exploitation strategy? Kernel regression does not provide an estimate of the uncertainty at the prediction. This is essential as the UCB strategy to pick the next point trades off exploration (uncertainty) vs. exploitation (certainty).

3. (5) You are using Bayesian Global Optimization to optimize the hyper-parameters of an SVM (σ and C) with Upper Confidence Bound (UCB). After a small number of training points have been sampled your model predicts that the setting σ_m, C_m yields the highest predicted accuracy on the validation set. Explain a scenario where your optimization algorithm would pick a different set of hyper-parameters σ', C' to explore with lower predicted accuracy.
   The prediction at the other point pair σ', C' might have very high variance and UCB picks points based on an additive combination of mean prediction and variance/standard deviation.

5 [19] Unsupervised Learning

1. (4) Batman wants to use $k-$means to cluster a data set of n data points. He doesn't know how many clusters k the data set has so he tries every possible setting $k = 1, ..., n$ and computes the squared reconstruction loss on his training data for each one. Then his algorithm selects the clustering with the lowest reconstruction error and outputs it as the final answer. After the algorithm is done he is amazed to note that the best reconstruction error is in fact *zero*. He sees this as a clear indication that his data has strong cluster structure, which he has now uncovered. Do you share his opinion? What value of k did the algorithm select?

2. (5) In the K-means optimization algorithm, we use the following update rule for each cluster where $\vec{\mu}_i$ is the center of the i-th cluster.

$$\gamma_{ij} = \begin{cases} 1 & : j = \min_j (\vec{x}_i - \vec{\mu}_j)^2 \\ 0 & : \text{else} \end{cases}$$

$$\vec{\mu}_j = \frac{1}{\sum_i \gamma_{ij}} \sum_i \gamma_{ij} \vec{x}_i$$

Prove that this minimizes the squared Euclidean distance between each point $\vec{x}_i$ and its cluster center $\vec{\mu}_j$.

3. (5) James Bond collects facial images of all the bad guys and good guys form all his past movies to train a good guy/bad guy classifier. He manages to get the images well aligned (each with black background). Unfortunately most of the old movies were made in the 1960s and are of pretty bad quality (assume additive Gaussian noise on the input pixels). How could he use PCA to remove this noise? How would he know what is data and what is noise?
   He applies PCA i.e. first subtracts the mean then compute the eigenvectors and eigenvalues of the covariance matrix. If he plots the eigenvalues in decreasing order he will see an elbow shaped curve and he can cut off right at the elbow. Alternatively, he could keep a high fraction (95% or 99%) of the variance by cutting off after m dimensions such that $\sum_{i=1}^{m} \lambda_i / \sum_{i=1}^{n} \lambda_i < 0.95$.

4. (5) Being a butterfly enthusiast, he also collects a data set of images of butterflies. Unfortunately he accidentally mixes it up with his data set of facial images. Given the PCA projection matrix (obtained from facial images) $\mathbf{U}$ and the mean face $\vec{m}$, how can he identify which images $\vec{x}_i$ are butterflies and which faces (without looking at the images - after all he can't see very well because he is wearing that ridiculous helmet with only tiny see-through slits for the eyes)?