Problem 1

In Neural Network Learning (Week 5 in Coursera) we asked you to compute the backpropagation of a model with two layers. At each layer, we used the sigmoid function as our activation function. Those activations were then used as an input to the next layer. The architecture of the model you will be working with is defined below. You have 20 training examples and 5 features.

- Input layer
- Hidden layer 1 (dim = 3)
- Hidden layer 2 (dim = 4)
- Output layer (dim = 2)

*Note 1:* Use the same notation as defined in section. \( \sigma \) denotes the sigmoid function.

*Note 2:* Consider a single training example input \( x \), represented as a column vector. The associated feed forward notation is \( z = Wx + b \) (where \( W \) is your weight matrix, corresponding to the \( \Theta \) matrix without the bias).
1. (1 point) What is the dimension of $X$, the matrix that stacks example inputs (excluding bias), that you are going to feed into your neural network?

2. (3 points) What are the dimensions of $W_1$, $W_2$, $W_3$ (do not include the bias)?

3. (3 points) What are the dimensions of $b_1$, $b_2$, $b_3$ (follow the method explained in section)?

4. (1 point) How many parameters are we training in our model? 
*Hint:* do not forget the bias!

5. (2 points) Using the same notations as in section, what are the dimensions of $a_1$, $a_2$ (including bias), where $a$ corresponds to the input of the next layer.

6. (4 points) Write the feed-forward equations of this neural network for a single training example vector $x$, from the beginning to the cost function $J$. Make sure your dimensions match. No coding! 
*Note:* assume that the activation function is the sigmoid, $\sigma$, for the first and second layer. For the last layer, use a softmax function. 
*Hint:* do not forget the bias!

7. So far, we only used the sigmoid function as an activation function. However, there are other popular functions.

   (a) (2 points) A similar function to the sigmoid is the hyperbolic tangent: 
   \[
   \tanh: x \mapsto \frac{e^x - e^{-x}}{e^x + e^{-x}}
   \]
   Prove that tanh is bounded by 1, i.e. 
   \[
   \forall x, |\tanh(x)| < 1
   \]

   (b) (3 points) Compute the derivative of tanh. We proved that the derivative of the sigmoid, $\sigma$, is $\sigma(1 - \sigma)$. In a similar way, you should be able to find an expression of the derivative of tanh that only depends on tanh.

   (c) (10 points) You already computed backpropagation using the sigmoid function in previous assignments/section. This time, let’s use hyperbolic tangent as the activation function in the two hidden layers. Write the vectorized backpropagation equations of this modified neural network, for a single training example vector $x$. Specifically, compute $\frac{\partial J}{\partial W_3}$, $\frac{\partial J}{\partial W_2}$, $\frac{\partial J}{\partial W_1}$. Mention explicitly the vectorized differentiation rules that you use and how you use them. 
*Note:* $\frac{\partial J}{\partial W}$ is defined as in section, it has the same shape as $W$. 
*Hint:* we still use the softmax for the last layer (so there is nothing to modify in that step).

8. (1 point) Why do not you regularize the bias term when regularizing the neural network?

**Problem 2: Bias-Variance trade-off**

1. (5 points) You have a neural network that predicts really well on the validation set. How does increasing the number of hidden units affect the training error? The validation error? Why?

2. (1 point) Remember that the regularized form of a cost function is: 
   \[
   \text{Regularized Cost} = \text{Cost} + \lambda \times \text{Penalty}, \; \lambda > 0
   \]
   How does the regularization penalty change with the value of $\lambda$?

3. (4 points) You fit a logistic regression for a classification problem. The training error is low but the validation error is high. How does regularizing affect the variance? The bias? What can you say about the training/validation error?

4. (5 points) Adding regularization creates a new hyperparameter: $\lambda$. Give one method to tune $\lambda$. 
*Note:* there are several.
Problem 3: Principal Component Analysis

1. Choose True/False. No justification needed.
   (a) (2 points) The goal of PCA is to interpret the underlying structure of the data in terms of the principal components that are best at predicting the output variable.
   (b) (2 points) The sum of the PCA eigenvalues is equal to the sum of the variances of the variables.
   (c) (2 points) Principal component analysis (PCA) can be used with variables of any mathematical types: quantitative, qualitative, or a mixture of these types.

2. Choose True/False. No justification needed.
   Remember that PCA is computed as follows:
   Step 1: Compute the covariance matrix: \( \Sigma = \frac{1}{m}X^TX \)
   Step 2: Compute the SVD of \( \Sigma \): \([U, S, V] = \text{SVD}(\Sigma)\)
   Step 3: Compute \( U_{\text{reduce}} = U[:, 1 : k] \) with \( k \) the number of principal components chosen
   Step 4: Compute the projections: \( Z = XU_{\text{reduce}} \)
   (a) (2 points) Removing columns of \( U \) will still result in an approximation of \( X \), but this will never be better than \( X \).
   (b) (2 points) In the case where \( U_{\text{reduce}} = U \), then \( ZZ^T = XX^T \). Hint: think about the properties of \( U \).

3. Given a data set, explain how you would use PCA. Specifically, answer these five questions:
   (a) (1 point) Why would you like to use feature normalization?
   (b) (1 point) What do the first two matrices of SVD represent?
   (c) (1 point) What do the eigenvectors represent?
   (d) (1 point) What do the eigenvalues represent?
   (e) (1 point) How do you choose the number of principal components?

4. (2 points) Give one advantage and one disadvantage of using PCA.

5. (3 points) How can PCA be used to speed up supervised learning?

Problem 4: Support Vector Machine

1. One of the most used kernels in SVM is the Gaussian RBF kernel:
   \[ k(x_i, x_j) = e^{-\frac{||x_i - x_j||^2}{2\sigma^2}} \]
   Prove the following properties:
   (a) (1 point) \( k \) is symmetric, i.e. \( \forall i,j \ k(x_i, x_j) = k(x_j, x_i) \)
   (b) (1 point) \( k \) is bounded, i.e. \( \forall i,j \ k(x_i, x_j) \leq 1 \)
   (c) (1 point) \( k \) is a similarity, i.e. \( \forall i,j \ k(x_i, x_j) = 1 \iff x_i = x_j \)
2. (2 points) Suppose we have three points $z_1$, $z_2$, $x$ such that $z_1$ is very close to $x$ (euclidean-distance), and $z_2$ is very far from $x$ (euclidean-distance). What can you say about $k(z_1, x)$ and $k(z_2, x)$? Choose one of the following. No justification needed.

(i) $k(z_1, x)$ will be close to 1 and $k(z_2, x)$ will be close to 0
(ii) $k(z_1, x)$ will be close to 0 and $k(z_2, x)$ will be close to 1
(iii) $k(z_1, x)$ will be close to $c_1$, s.t. $c_1 > 1$ and $k(z_2, x)$ will be close to $c_2$ s.t. $c_2 < 0$
(iv) $k(z_1, x)$ will be close to $c_1$, s.t. $c_1 < 0$ and $k(z_2, x)$ will be close to $c_2$ s.t. $c_2 > 1$

3. (5 points) When would you like to use a SVM? (1 case) Why? (2 reasons)

4. (2 points) Assume you are building a SVM model on a one vs. all classification problem with 3 classes. How many times do you have to run a SVM?

5. (3 points) What is the point of using a kernel in SVMs? How do these kernels work?

**Problem 5: K-means algorithm**

Suppose we have the following points in one dimension:

$$x_1 = 0, x_2 = 2, x_3 = 3, x_4 = 8, x_5 = 10$$

Run the 2-means clustering until convergence with the following initialization:

$$\mu_1 = -1, \mu_2 = 5$$

Note: in the case of a tie, assign the point to the class with a lower number (i.e. if one point is tied between class 1 and class 2, assign it to class 1).

1. (2 points) Draw a number line to help you visualize what is happening.

2. (2 points) How many iterations did you perform? *Note*: do not double count! Therefore if iterations $n$ and $n + 1$ give the same result, the algorithm converges in $n$ iterations.

3. (5 points) What is the final assignment?

4. (5 points) What are the final centroids?

5. (5 points) Remember that the loss in the K-means algorithm is given by:

$$\text{Loss} = \sum_{i=1}^{m} ||x_i - \mu_{z_i}||^2$$

with $z_i$ the cluster of point $i$.

Compute the final loss.

6. (1 point) In the general case, does the K-means algorithm converge to the global minimum?