Machine Learning *CS 4364/6364, Fall 2016*

HW3

Instructions

Collaboration: You are allowed to discuss the problems with other students. However, you must write up your own solutions for the math questions, and implement your own solutions for the programming problems. Please list all collaborators and sources consulted at the top of your homework.

Submission: Please submit homework pdf's at this address: machine.learning.gwu 'at' gmail.com. Electronic submissions are preferred. Math can be formatted in Latex (some easy editors for Latex are Lyx, TexShop), Word, or other editors. Math solutions can be optionally submitted on paper at the Department of Computer Science in person (or in the dropbox to the right of the entrance to SEH 4000 if submitted past 5 pm), but all code should be submitted electronically. The assignment is due Friday, November 4, at 5 pm.

Questions

1. **Kernels and Maximum Margin Classifiers - 10 pts**

    1. We have discussed two different definitions of kernels in class
        - Definition 1: $K(x, x')$ is a kernel if it can be written as an inner product $\phi(x)^T\phi(x')$ for some feature mapping $x \to \phi(x)$.
        - Defintion 2: $K(x, x')$ is a kernel if for any finite set of training examples, $x_1, \ldots, x_n$, the $n \times n$ matrix $\boldsymbol{K}$ such that $K_{ij} = K(x_i, x_j)$ is positive semidefinite.

        Show that Definition 1 implies Definition 2.

        Hint: you could show this by proving that for any real numbers $\alpha_1, \ldots, \alpha_n$ and points $x_1, \ldots, x_n$,

        $$\Sigma_{i=1}^n \Sigma_{j=1}^n \alpha_i \alpha_j K(x_i, x_j) \geq 0$$

        if the kernel can be written as $K(x, x') = \phi(x)^T\phi(x')$.

    2. One way to construct kernels is to build them from simpler ones. We have already seen three possible "composition rules": assuming $K_1(x, x')$ and $K_2(x, x')$ are kernels then so are
        - (scaling) $f(x)K_1(x, x')f(x'), f(x) \in \mathbb{R}$
        - (sum) $K_1(x, x') + K_2(x, x')$
        - (product) $K_1(x, x')K_2(x, x')$

(a) Let $\phi^{(1)}(x)$ and $\phi^{(2)}(x)$ be the feature vectors corresponding to kernels $K_1(x, x')$ and $K_2(x, x')$, respectively. These feature vectors may be of different length. Show that the product $K_1(x, x')K_2(x, x')$ is a kernel by showing that its feature vectors are given explicitly by $\phi(x)$ whose $(i, j)^{th}$ component (doubly indexed vector) is $\phi_i^{(1)}(x)\phi_j^{(2)}(x)$.

(b) Use the composition rules to build a normalized cubic polynomial kernel

$$K(x, x') = \left(1 + \left(\frac{x}{||x||}\right)^T \left(\frac{x'}{||x'||}\right)\right)^3$$

You can assume that you already have a constant kernel $K_0(x, x') = 1$ and a linear kernel $K_1(x, x') = x^T x'$. Identify which rules you are employing at each step.

3. **[Undergrads can optionally submit this part for extra credit]** Let's now explore the effect of feature vectors on the maximum margin solution. Consider a simple one dimensional case where we have only two training examples ($x_1 = 0$, $y_1 = -1$), ($x_2 = \sqrt{2}$, $y_2 = 1$), and we map each input to a feature vector $\phi(x) = \begin{bmatrix}1 & \sqrt{2}x & x^2\end{bmatrix}^T$. In other words, we are effectively using a second order polynomial kernel. We'd like to find and understand the maximum margin solution $\hat{\boldsymbol{w}} = \begin{bmatrix}\hat{w}_1 & \hat{w}_2 & \hat{w}_3\end{bmatrix}^T$ and $\hat{w}_0$ to

$$\begin{gathered}\min ||\boldsymbol{w_1}||^2 \text{ subject to} \\ y_1[w_0 + \phi(x_1)^T \boldsymbol{w_1}] - 1 \geq 0 \\ y_2[w_0 + \phi(x_2)^T \boldsymbol{w_1}] - 1 \geq 0\end{gathered}$$

Please return your derivations along with the specific answers.

(a) Using your knowledge of the maximum margin boundary, write down a vector that points in the same direction as $\hat{\boldsymbol{w}}_1$

(b) What is the value of the margin that we can achieve in this case?

(c) By relating the margin and $\hat{\boldsymbol{w}}_1$, provide the actual solution $\hat{\boldsymbol{w}}_1$. What is $\hat{w}_0$?

2. **Gaussian Mixtures, Logistic and Softmax models - 10 pts**

You are provided with the following function: `plotMixtureGaussians.m`. Given a mixture of two Gaussians parameterized by $\theta = \{\mu_1, \Sigma_1, \mu_2, \Sigma_2, w\}$, this function plots the Gaussians and the associated decision boundary.

(a) Use this function to experiment with different Gaussians and their decision boundaries. For each of the following types of decision boundaries, find a set of parameters that result in the specified shape. Turn in the plot and the full specification of the Gaussian Mixtures (the means, covariance matrices, and weights). Briefly justify your choice of parameters.

(i) A linear decision boundary between the means of the two Gaussians.

(ii) A linear decision boundary where both means are on the same side of the decision boundary.

(iii) A non-continuous decision boundary (i.e. one of the classes is represented by two disconnected regions).

(iv) A circular decision boundary.

(v) No decision boundary - the entire plane is one decision region.

(b) In many cases, it is necessary to classify into more than two classes. A natural extension of the Gaussian mixture approach is to fit a Gaussian distribution for each class, then classify each input vector to the class that results in the highest posterior probability.

Another possible approach is to generalize the the logistic regression model to multiple classes. Let $x = [x_1, x_2, ..., x_d]^T$ be an input vector, and suppose we would like to classify it to k possible classes. That is, the output y can take a value in $\{1, ..., k\}$. The $softmax$ generalization of the logistic model uses $k(d+1)$ weights $w = (w_{ij})$, $i = 1, ..., k$, $j = 0, ..., d$, which define the following k intermediate values:

$$\begin{aligned} z_1 &= w_{10} + \sum_j w_{1j} x_j \\ &... \\ z_i &= w_{i0} + \sum_j w_{ij} x_j \\ &... \\ z_k &= w_{k0} + \sum_j w_{kj} x_j \end{aligned}$$

The softmax probabilities are given by

$$Pr(y = i|x) = \frac{\exp(-z_i)}{\sum_{j=1}^{k} \exp(-z_j)}$$

Show that when $k = 2$ the $softmax$ model reduces to the logistic regression model. That is, show how both give rise to the same classification probabilities $Pr(y|x)$. Do this by constructing an explicit transformation between the weights: for any given set of $2(d+1)$ $softmax$ weights, show an equivalent set of $(d+1)$ logistic weights. (An interesting side note : It can be shown that the softmax model, for any k, can always be represented by a Gaussian mixture model.)

3. **Support Vector Machines - MATLAB Programming - 10 pts**
In this problem, we use the IRIS dataset discussed in class to construct two binary classification problems.

- P1 : A binary classification problem to determine whether a given flower is a setosa or not. You can find the associated files in the folder iris_1
- P2 : A binary classification problem to determine whether a given flower is a virsicolor or not. You can find the associated files in the directory iris_2.
- P3 : A binary classification problem to determine whether a given flower is a virsicolor or not. However, instead of all 4 features, we use only the two features used in P1. You can find the associated files in the directory iris_3. This makes it easier to visualize the results.

For this question, use the `svmtrain` and `svmclassify` functions in Matlab.

(a) Consider the classification problem P1. Using the training data and labels, determine and plot the classification boundary (you just need the support vectors in this case) using the `svmtrain` function. Note, the plot can be obtained by passing an additional parameter to the `svmtrain` function called `showplot`. For details, refer to the help page of `svmtrain`.

Next, use the test data provided to determine the error rate for this classifier. You can use the `svmclassify` function to do this. Also plot the result of classification using the `showplot` option for `svmclassify`.

(b) Now that you are familiar with `svmtrain` and `svmclassify`, let us consider the problem P2. It is known that this problem is not linearly separable. Hence, for this problem, you are asked to experiment with the following kernels.

- Linear : This is the default option for `svmtrain`.
- A polynomial kernel of degree 3.
- A Gaussian Radial Basis Function with scaling factor 1.

Use the default setting of the parameter C ("box constraint"). Train using the training data and report the error rate on the test data for each of the kernels above.

(c) `Kernel Perceptron` : Implement the Kernel Perceptron as described in lecture using each of the kernel functions mentioned in the previous problem. Use the training and test data provided for the problem P3 to train and test your implementation. Report the error rate for each kernel function.

(d) Here, you'll compare discriminative learning and generative learning. Use the code for EM from Homework 2 and the training data for P3 to obtain a mixture of two Gaussians. Write Matlab code that takes a mixture of 2 Gaussians (one for each label) and an unlabeled data point, and returns a label for the data point. Use this function to determine the error rate on the test data. Using the same training and test sets (P3), build an SVM classifier using `svmtrain` and determine its error rate using `svmclassify`. (You may experiment with different kernels for this problem, but use the default C.)