CSE 512 Machine Learning: Homework 3

Department of Computer Science
Stony Brook University

- There are 4 questions on this assignment. The first question involves coding. Do *not* attach your code to the writeup. Instead, zip and submit your code electronically on Blackboard (Bb). Name your .zip file as **[your_SBU_name].zip**, e.g. vivek.zip
- The assignment is due at 5:30 PM (beginning of class) on **Tuesday April 14, 2015**.
- Do not forget to put both your name and SBU ID on *each* page of your submission.
- If you have any questions, please direct your question first to the TA, then the instructor.
- You may *discuss* the questions with fellow students, *however* you must always write your own solutions and must acknowledge whom you discussed the question with. Do not copy from other sources, share your work with others, or search for solutions on the web.

1 k-NN, SVM, and Cross-Validation [40 points]

In this question, you will explore how cross-validation (CV) can be used to fit "magic parameters". More specifically, you'll fit the constant k in the k-Nearest Neighbor algorithm (k-NN), and the slack penalty C in the case of Support Vector Machines (SVM). For *all* implementation questions, please submit your source code as outlined in the instructions and **supply pseudo-code** in your writeup where requested.

Dataset

Download the data file available from `http://www.cs.stonybrook.edu/~leman/courses/15CSE512/hws/hw3_matlab.zip` and unpack it. The file `faces.mat` contains the Matlab variables `traindata` (training data), `trainlabels` (training labels), `testdata` (test data), `testlabels` (test labels) and `evaldata` (evaluation data, needed later).

This is a facial attractiveness classification task: Given a picture of a face, you need to predict whether the rating of the face is *hot* or *not*. So, each row corresponds to a data point (a picture). Each column is a feature, a pixel. The value of the feature is the intensity of the pixel in the grayscale image. (This is an "easier" version of the dataset [1] on the project website.) For fun, try `showface(evaldata(1,:))`, `showface(evaldata(2,:))`.

`cosineDistance.m` implements *cosine distance*, a simple distance function between two feature vectors $\mathbf{x}$ and $\mathbf{y}$. You will use this distance function as the metric for your k-Nearest Neighbor classifier.

1.1 k-NN (20 pts)

1. [10pt] Implement the k-Nearest Neighbor (k-NN) algorithm using the given distance function. Outline the pseudo-code in your hard copy.
   Hint: You might want to precompute the distances between all pairs of points, to speed up the cross-validation later.

[1] Ryan White, Ashley Eden, Michael Maire "Automatic Prediction of Human Attractiveness", December 2003.

2. [5pt] Implement n-fold cross validation for k-NN. Your implementation should partition the training data and labels into n parts of approximately equal size. Outline the pseudo-code in your hard copy.

3. [5pt] For $k = 1, 2, \ldots, 100$, compute and plot the 10-fold (i.e., $n = 10$) cross-validation error for the training data, the training error, and the test error. Don't forget to hand in the plot!

   How do you interpret these plots? Does the value of k that minimizes the cross-validation error also minimize the test set error? Does it minimize the training set error? Either way, can you explain why? Also, what does this tell us about using the training error to pick the value of k?

1.2 SVM (20pts)

1. [5pt] Now download *libsvm* provided at `http://www.cs.stonybrook.edu/~leman/courses/15CSE512/hws/libsvm-mat-2.89-3.zip` and unpack it to your working directory. It has a Matlab interface which includes binaries for Windows. It can be used on OS X or Unix but has to be compiled (requires `g++` and `make`) – see the `README` file from the *libsvm* zip package and/or the instructions at `http://www.cs.stonybrook.edu/~leman/courses/15CSE512/hws/libsvm_inst.txt`.

   `hw3_matlab.zip`, which you downloaded earlier, contains files `testSVM.m` (an example demonstration script), `trainSVM.m` (for training) and `classifySVM.m` (for classification), which will show you how to use *libsvm* for training and classifying using a SVM. Run `testSVM`. This should report a test error of 0.4333.

   In order to train a SVM with slack penalty C on training set `data` with labels `labels`, call
   ```
   svmModel = trainSVM(data, labels, C)
   ```
   In order to classify examples `test`, call
   ```
   testLabels = classifySVM(svmModel, test)
   ```
   Train a SVM on the training data with $C = 500$, and report the error on the test set.

2. [10pt] Now implement n-fold cross-validation for SVMs. Similar to k-NN, split your training data into n roughly equal parts. Hand in the pseudo-code in your hard copy.

3. [5pt] For $C = 10, 10^2, 10^3, 10^4, 5 \cdot 10^4, 10^5, 5 \cdot 10^5, 10^6$, compute and plot the 10-fold (i.e., $n = 10$) cross-validation error for the training data, the training error, and the test error, with the axis for C in log-scale (try `semilogx`). Don't forget to hand in the plot!

   How do you interpret these plots? Does the value of C which minimizes the cross-validation error also minimize the test set error? Does it minimize the training set error? Either way, can you explain why? Also, what does this tell us about using the training error to pick the value of C?

2 Feature Maps, Kernels, and SVM [20 points]

You are given a data set D in Figure 1 with data from a single feature X_1 in $\mathbb{R}^1$ and corresponding label $Y \in \{+, -\}$. The data set contains three positive examples at $X_1 = \{-3, -2, 3\}$ and three negative examples at $X_1 = \{-1, 0, 1\}$.

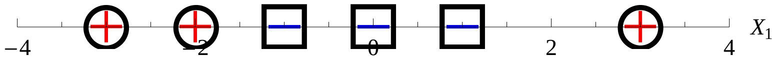


Figure 1: Dataset for SVM feature map task in Question 2.

2.1 Finite Features and SVMs

1. [1pt] Can this data set (in its current feature space) be perfectly separated using a linear separator? Why or why not?

2. [1pt] Lets define the simple feature map $\phi(u) = (u, u^2)$ which transforms points in $\mathbb{R}^1$ to points in $\mathbb{R}^2$. Apply ϕ to the data and plot the points in the new $\mathbb{R}^2$ feature space.

3. [1pt] Can a linear separator perfectly separate the points in the new $\mathbb{R}^2$ features space induced by ϕ? Why or why not?

4. [1pt] Give the analytic form of the kernel that corresponds to the feature map ϕ in terms of only X_1 and X_1'. Specifically define $k(X_1, X_1')$.

5. [3pt] Construct a maximum-margin separating hyperplane for the feature mapped data set. This hyperplane will be a line in $\mathbb{R}^2$, which can be parameterized by its normal equation, i.e. $w_1 Y_1 + w_2 Y_2 + c = 0$ for appropriate choices of w_1, w_2 and c. Here, $(Y_1, Y_2) = \phi(X_1)$ is the result of applying the feature map ϕ to the original feature X_1. Give the values for w_1, w_2 and c. Also, explicitly compute the margin for your hyperplane. You do not need to solve a quadratic program to find the maximum margin hyperplane. Note that the hyperplane must pass somewhere between (-2,4) and (-1,1) (Why?), and that the hyperplane must be perpendicular to the line connecting these two points. Use only two support vectors.

6. [1pt] On the plot of the transformed points (from part 3), plot the separating hyperplane and the margin, and circle the support vectors.

7. [1pt] Draw the decision boundary separating of the separating hyperplane, in the original $\mathbb{R}^1$ feature space.

8. [3pt] Observe that we can formulate a SVM as learning a linear classifier given by Equation 1 where the optimization problem is defined over α and outlined by the dual form formulation discussed in class. Compute the coefficients of α and the constant b in Equation 1 for the kernel k and the support vectors $SV = \{u_1, u_2\}$ you chose in part 6. Be sure to explain how you obtained these coefficients.

$$y(x) = \mathsf{sign}\left(\sum_{n=1}^{|SV|} \alpha_n y_n k(x, u_n) + b\right) \tag{1}$$

   Think about the dual form of the quadratic program and the constraints placed on the components of α.

   Hint: Note that the support vectors which we denote by $SV = \{u_1, u_2\}$ correspond to exactly those training data points which lie on the margin.

9. [1pt] If we add another positive ($Y = +$) point to the training set at $X_1 = 5$ would the hyperplane or margin change? Why or why not?

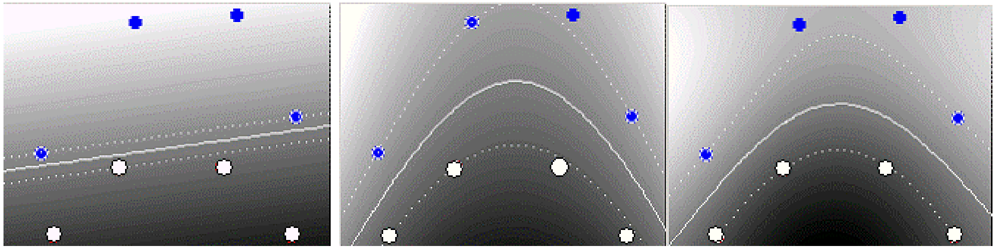
10. [3pts] Three different support vector machines have been trained on a 2D data set using:

    (a) a linear kernel, $k(\mathbf{x}, \mathbf{y}) = \mathbf{x}^T \mathbf{y}$ (Figure 2a)

    (b) a quadratic polynomial kernel, $k(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^T \mathbf{y} + 1)^2$ (Figure 2b)

    (c) an RBF kernel, $k(\mathbf{x}, \mathbf{y}) = exp\big(-\frac{1}{2\sigma}(\|\mathbf{x} - \mathbf{y}\|)^2\big)$ (Figure 2c)

    Assume we now translate the data by adding a large constant value (i.e. 10) to the vertical coordinate of each of the data points, i.e. a point (x,y) becomes (x,y+10).



(a) Linear kernel (b) Quadratic polynomial kernel (c) RBF kernel

Figure 2: Three different kernels are trained on a 2D data set used in problem 2.1.10

If we retrain the above SVMs on this new data, do the resulting SVM boundary change relative to the data points? Say if it does change or not for case (a), case (b) and case (c). Briefly explain why or why not, it changes for all 3 cases (a), (b) and (c) and suggest or draw what happens to the resulting new boundaries when appropriate.

2.2 Infinite Features Spaces and Kernel Magic

Lets define a new (infinitely) more complicated feature transformation $\phi_n : \mathbb{R}^1 \to \mathbb{R}^n$ given in Equation 2.

$$\phi_n(x) = \left\{ e^{-x^2/2}, e^{-x^2/2}x, \frac{e^{-x^2/2}x^2}{\sqrt{2}}, \ldots, \frac{e^{-x^2/2}x^i}{\sqrt{i!}} \ldots, \frac{e^{-x^2/2}x^n}{\sqrt{n!}} \right\} \tag{2}$$

Suppose we let $n \to \infty$ and define a new feature transformation in Equation 3. You can think of this feature transformation as taking some finite feature vector and producing an infinite dimensional feature vector rather than the simple two dimensional feature vector used in the earlier part of this problem.

$$\phi_\infty(x) = \left\{ e^{-x^2/2}, e^{-x^2/2}x, \frac{e^{-x^2/2}x^2}{\sqrt{2}}, \ldots, \frac{e^{-x^2/2}x^i}{\sqrt{i!}} \ldots \right\} \tag{3}$$

1. [3pt] We know that we can express a linear classifier using only inner products of support vectors in the transformed feature space as seen in Equation 1. It would be great if we could some how use the feature space obtained by the feature transformormation ϕ_∞. However, to do this we must be able to compute the inner product of examples in this infinite vector space. Lets define the inner product between two infinite vectors $\mathbf{p} = \{p_1, \ldots, p_i, \ldots\}$ and $\mathbf{q} = \{q_1, \ldots, q_i, \ldots\}$ as the infinite sum given in Equation 4.

$$< \mathbf{p}, \mathbf{q} >= \mathbf{p} \cdot \mathbf{q} = \sum_{i=1}^{\infty} p_i q_i \tag{4}$$

Based on this, can we explicity compute $k(a,b)$? Note that $k(a,b)$ computes the inner product when a and b are mapped to the infinite dimension vector space without having to explicitly map a and b to this infinite dimension vector space. What is the explicit form of $k(a,b)$? *Hint*: You may want to use the Taylor series expansion of e^x which is given in Equation 5.

$$e^x = \lim_{n\to\infty} \sum_{i=0}^{n} \frac{x^i}{i!} \tag{5}$$

2. [1pt] With such a high dimensional feature space should we be concerned about overfitting?

3 Learning Theory [20 points]

3.1 VC Dimension

In this section you will calculate the VC-dimension of some hypothesis classes. Remember that in order to prove that H has VC-dimension d you need to show that

- There exists a set of d points which can be shattered by H. (This step is often easy.)
- There exists **no** set of $d+1$ points that can be shattered by H. (This step is hard.)

1. (5 points) Find the VC-dimension of the hypothesis class that consists of the union of k intervals on the real line. In other words each hypothesis $h \in H$ is associated with k closed intervals $[a_i, b_i],\ i \in \{1, 2, \ldots, k\}$; and $h(x) = 1$ iff $x \in \cup_{i \in \{1,2,\ldots,k\}} [a_i, b_i]$.
2. (5 points) Consider the hypothesis class of linear classifiers with offset in d dimensions:

$$\mathcal{H} = \{\text{sign}(\theta \cdot \mathbf{x} + \theta_0) : \theta \in R^d, \theta_0 \in R\}$$

   Show that there exists a set of $d+1$ points $\{\mathbf{x}_0, \mathbf{x}_1, \ldots, \mathbf{x}_d\}$ that can be shattered by $\mathcal{H}$. Specifically, first specify the points, and then given any labeling $y_0, y_1, \ldots, y_d$, describe explicitly how to construct a classifier in $\mathcal{H}$ that agrees with the labeling.

3. (5 points) Let C be a finite hypothesis class that is, $|C| < \infty$. Show that the VC-dimension of C is bound: $\text{VCDim}(C) \leq \log_2 |C|$.

3.2 Sample Complexity

In this part, you will use sample complexity bounds to determine how many training examples are needed to find a good classifier.

- (5 points) Let H be the hypothesis class of linear separators. Recall that the VC dimension of linear separators in $\mathbb{R}^n$ is $n+1$. Suppose we sample a number m of training examples i.i.d. according to some unknown distribution $\mathcal{D}$ over $\mathbb{R}^2 \times \{-1, 1\}$.

$$(X_1, Y_1), (X_2, Y_2), \ldots, (X_m, Y_m) \sim \mathcal{D}$$

  Prove that if $m \geq 14619$, then with probability at least .99 over the draw of the training examples, the linear separator with smallest training error $\hat{h}_{\text{ERM}} = \arg\min_{h \in H} \text{error}_{train}(h)$ has

$$\text{error}_{true}(\hat{h}_{\text{ERM}}) - \text{error}_{train}(\hat{h}_{\text{ERM}}) \leq .05$$

  You may *not* assume $\text{error}_{train}(\hat{h}_{\text{ERM}}) = 0$. You may use any formulas from the lecture slides, textbook, or readings from the website, but please tell us where you found the formula(s) you use.

4 Bayes Nets [20 points]

Consider the Bayes Net depicted in Figure 3 for modeling system safety. Assume each random variable X_i is a boolean random variable.

1. (2 point) How many parameters do you need to estimate to compute the full joint distribution of the 9 variables associated with system safety *without* any independence assumptions ?

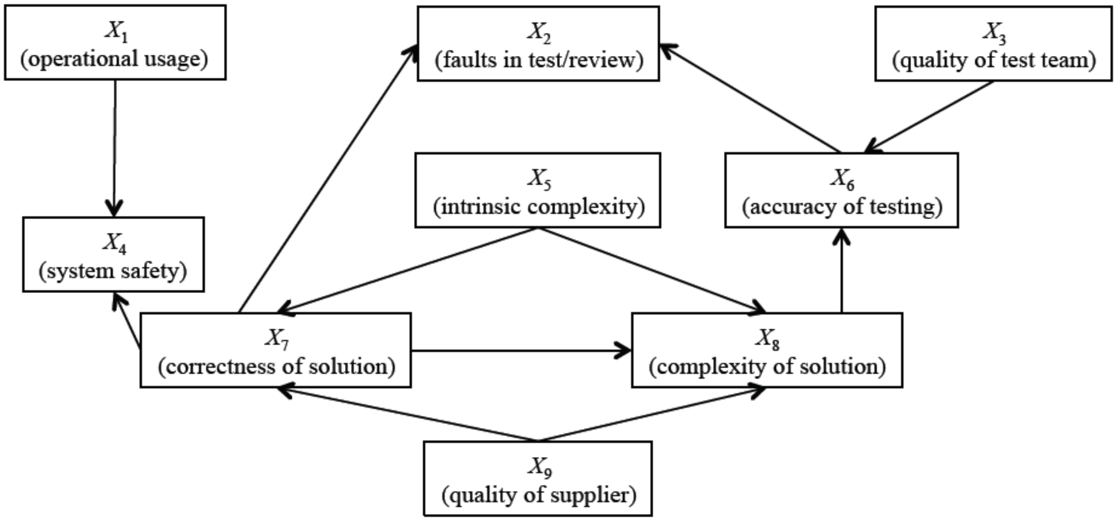


Figure 3: Bayes Net for system safety.

2. (3 points) How many parameters do you need to estimate to compute the full joint distribution assuming the Bayesian Network depicted ?
3. (5 points) Is X_5 conditionally independent of X_6 given X_8 ?
4. (2 points) Is X_5 conditionally independent of X_6 given X_2 ? Briefly explain your reasoning.
5. (3 points) What is the minimal set of nodes which when observed renders X_8 conditionally independent of all other nodes ?
6. (5 points) We would like to compute the marginal probability for X_9. Which elimination order should we use to minimize the number of additions ?