10-701 Midterm Exam Solutions, Spring 2007

1. Personal info:
   - Name:
   - Andrew account:
   - E-mail address:
2. There should be 16 numbered pages in this exam (including this cover sheet).
3. You can use any material you brought: any book, class notes, your print outs of class materials that are on the class website, including my annotated slides and relevant readings, and Andrew Moore's tutorials. You cannot use materials brought by other students. Calculators are allowed, but no laptops, PDAs, phones or Internet access.
4. If you need more room to work out your answer to a question, use the back of the page and clearly mark on the front of the page if we are to look at what's on the back.
5. Work efficiently. Some questions are easier, some more difficult. Be sure to give yourself time to answer all of the easy ones, and avoid getting bogged down in the more difficult ones before you have answered the easier ones.
6. Note there are extra-credit sub-questions. The grade curve will be made without considering students' extra credit points. The extra credit will then be used to try to bump your grade up without affecting anyone else's grade.
7. You have 80 minutes.
8. Good luck!

Question	Topic	Max. score	Score
1	Short questions	21 + 0.911 extra	
2	SVM and slacks	16	
3	GNB	8	
4	Feature Selection	10	
5	Irrelevant Features	14 + 3 extra	
6	Neural Nets	16 + 5 extra	
7	Learning theory	15	

1 [21 Points] Short Questions

The following short questions should be answered with at most two sentences, and/or a picture. For the **(true/false)** questions, answer true or false. If you answer true, provide a short justification, if false explain why or provide a small counterexample.

1. [1 point] **true/false** A classifier trained on less training data is less likely to overfit.

   ★ **SOLUTION:** This is false. A specific classifier (with some fixed model complexity) will be more likely to overfit to noise in the training data when there is less training data, and is therefore more likely to overfit.

2. [1 point] **true/false** Given m data points, the training error converges to the true error as $m \to \infty$.

   ★ **SOLUTION:** This is true, if we assume that the data points are i.i.d. A few students pointed out that this might not be the case.

3. [1 point] **true/false** The maximum likelihood model parameters (α) can be learned using linear regression for the model: $y_i = \alpha_1 x_1 x_2^3 + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$ iid noise.

   ★ **SOLUTION:** This is true. y is linear in α_1, so it can be learned using linear regression.

4. [2 points] **true/false** The maximum likelihood model parameters (α) can be learned using linear regression for the model: $y_i = x_1^{\alpha_1} e^{\alpha_2} + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$ iid noise.

   ★ **SOLUTION:** This is false. y is not linear in α_1 and α_2, and no simple transformation will make it linear ($\log[x_1^{\alpha_1} e^{\alpha_2} + \epsilon_i] \neq \alpha_1 \log x_1 + \alpha_2 + \epsilon_i$).

5. [2 points] **true/false** The maximum likelihood model parameters (α) can be learned using linear regression for the model: $y_i = \log(x_1^{\alpha_1} e^{\alpha_2}) + \epsilon_i$ where $\epsilon_i \sim N(0, \sigma^2)$ iid noise.

   ★ **SOLUTION:** This is true. $y_i = \log(x_1^{\alpha_1} e^{\alpha_2}) + \epsilon_i = \alpha_1 \log x_1 + \alpha_2 + \epsilon_i$, which is linear in α_1 and α_2. Also, assuming $x_1 > 0$.

6. [2 points] **true/false** In AdaBoost weights of the misclassified examples go up by the same multiplicative factor.

   ★ **SOLUTION:** True, follows from the update equation.

7. [2 points] **true/false** In AdaBoost, weighted training error ϵ_t of the t^{th} weak classifier on training data with weights D_t tends to increase as a function of t.

   ★ **SOLUTION:** True. In the course of boosting iterations the weak classifiers are forced to try to classify more difficult examples. The weights will increase for examples that are repeatedly misclassified by the weak classifiers. The weighted training error ϵ_t of the t^{th} weak classifier on the training data therefore tends to increase.

8. [2 points] **true/false** AdaBoost will eventually give zero training error regardless of the type of weak classifier it uses, provided enough iterations are performed.

   ★ **SOLUTION:** Not if the data in the training set cannot be separated by a linear combination of the specific type of weak classifiers we are using. For example consider the EXOR example(In hw2 we worked with a *rotated* EXOR toy dataset) with decision stumps as weak classifiers. No matter how many iterations are performed zero training error will not be achieved.

9. [2 points] Consider a point that is correctly classified and distant from the decision boundary. Why would SVM's decision boundary be unaffected by this point, but the one learned by logistic regression be affected?

   ★ **SOLUTION:** The hinge loss used by SVMs gives zero weight to these points while the log-loss used by logistic regression gives a little bit of weight to these points.

10. [2 points] Why does the kernel trick allow us to solve SVMs with high dimensional feature spaces, without significantly increasing the running time?

★ **SOLUTION:** In the dual formulation of the SVM, features only appear as dot products which can be represented compactly by kernels.

11. [2 points] Consider a learning problem with 2D features. How are the decision tree and 1-nearest neighbor decision boundaries related?

    ★ **SOLUTION:** In both cases, the decision boundary is piecewise linear. Decision trees do axis-aligned splits while 1-NN gives a voronoi diagram.

12. [2 points] You are a reviewer for the International Mega-Conference on Algorithms for Radical Learning of Outrageous Stuff, and you read papers with the following experimental setups. Would you accept or reject each paper? Provide a one sentence justification. (This conference has short reviews.)

    - **accept/reject** "My algorithm is better than yours. Look at the training error rates!"

      ★ **SOLUTION:** Reject - the training error is optimistically biased.

    - **accept/reject** "My algorithm is better than yours. Look at the test error rates! (Footnote: reported results for $\lambda = 1.789489345672120002$.)"

      ★ **SOLUTION:** Reject - A λ with 15 decimal places suggests a highly tuned solution, probably looking at the test data.

    - **accept/reject** "My algorithm is better than yours. Look at the test error rates! (Footnote: reported results for best value of λ.)"

      ★ **SOLUTION:** Reject - Choosing λ based on the test data?

    - **accept/reject** "My algorithm is better than yours. Look at the test error rates! (Footnote: reported results for best value of λ, chosen with 10-fold cross validation.)"

      ★ **SOLUTION:** Accept - Cross validation is the appropriate method for selecting parameters.

13. [Extra credit: 0.911 points] You have designed the ultimate learning algorithm that uses physical and metaphysical knowledge to learn and generalize beyond the quantum P-NP barrier. You are now given the following test example:

    What label will your algorithm output?

(a) Watch a cartoon.

(b) Call the anti-terrorism squad.

(c) Support the Boston Red Sox.

(d) All labels have equal probability.

★ **SOLUTION:** Watching a cartoon earned you 0.39 points. 0.2005 points were given for supporting the Boston Red Sox. 0.666 points were given for calling the anti-terrorism squad. 0.911 points were given for "all labels have equal probability."

■ **COMMON MISTAKE :** Some students skipped this question; perhaps a Moonite Marauders is also scary on paper...

2 [16 Points] SVMs and the slack penalty C

The goal of this problem is to correctly classify test data points, given a training data set. You have been warned, however, that the training data comes from sensors which can be error-prone, so you should avoid trusting any specific point too much.

For this problem, assume that we are training an SVM with a **quadratic kernel**– that is, our kernel function is a polynomial kernel of degree 2. You are given the data set presented in Figure 1. The slack penalty C will determine the location of the separating hyperplane. Please answer the following questions *qualitatively*. Give a one sentence answer/justification for each and draw your solution in the appropriate part of the Figure at the end of the problem.

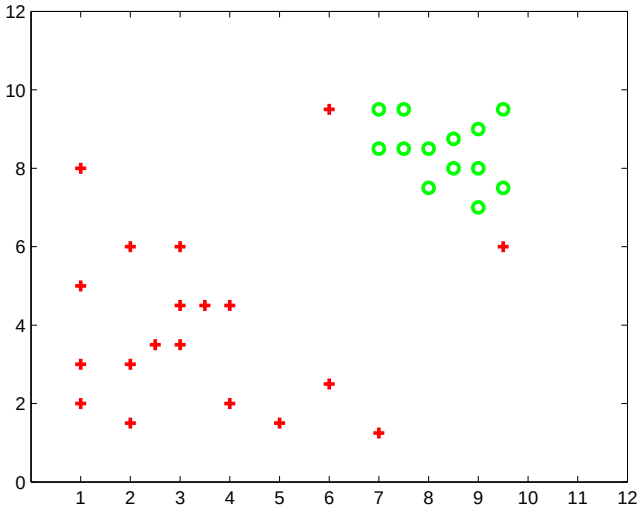


Figure 1: Dataset for SVM slack penalty selection task in Question 2.

1. [4 points] Where would the decision boundary be for very large values of C (i.e., $C \to \infty$)? (remember that we are using an SVM with a quadratic kernel.) Draw on the figure below. Justify your answer.

   ★ **SOLUTION:** For large values of C, the penalty for misclassifying points is very high, so the decision boundary will perfectly separate the data if possible. See below for the boundary learned using libSVM and $C = 100000$.

   ■ **COMMON MISTAKE 1:** Some students drew straight lines, which would not be the result with a quadratic kernel.

■ **COMMON MISTAKE 2:** Some students confused the effect of C and thought that a large C meant that the algorithm would be more tolerant of misclassifications.

2. [4 points] For $C \approx 0$, indicate in the figure below, where you would expect the decision boundary to be? Justify your answer.

★ **SOLUTION:** The classifier can maximize the margin between most of the points, while misclassifying a few points, because the penalty is so low. See below for the boundary learned by libSVM with $C = 0.00005$.

3. [2 points] Which of the two cases above would you expect to work better in the classification task? Why?

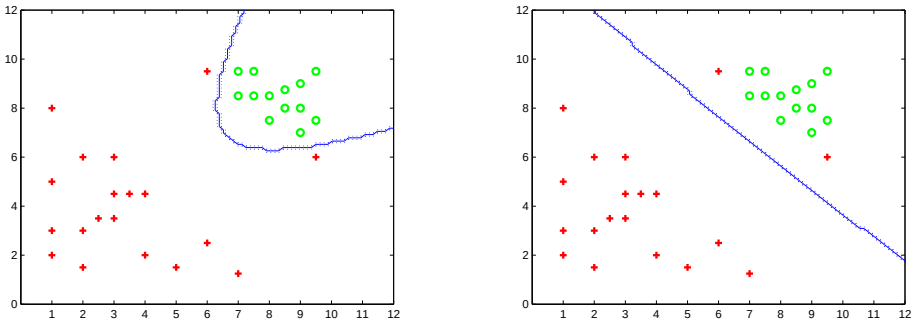
★ **SOLUTION:** We were warned not to trust any specific data point too much, so we prefer the solution where $C \approx 0$, because it maximizes the margin between the dominant clouds of points.

4. [3 points] Draw a data point which will not change the decision boundary learned for very large values of C. Justify your answer.

★ **SOLUTION:** We add the point circled below, which is correctly classified by the original classifier, and will not be a support vector.

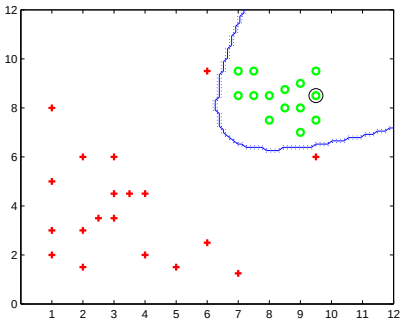
5. [3 points] Draw a data point which will significantly change the decision boundary learned for very large values of C. Justify your answer.

★ **SOLUTION:** Since C is very large, adding a point that would be incorrectly classified by the original boundary will force the boundary to move.

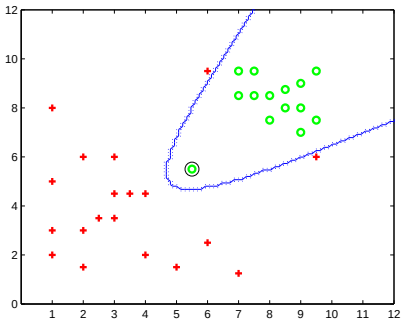


(a) Part 1

(b) Part 2



(c) Part 4



(d) Part 5

Figure 2: Solutions for Problem 2

3 [10 points] Feature selection with boosting

Consider a text classification task, such that the document X can be expressed as a binary feature vector of the words. More formally $X = [X_1, X_2, X_3, ..., X_m]$, where $X_j = 1$ if word j is present in document X, and zero otherwise. Consider using the AdaBoost algorithm with a simple weak learner, namely

$$h(X;\theta) = yX_j$$
$$\theta = \{j, y\} \; j \text{ is the word selector ; } y \text{ is the associated class}$$
$$y \in \{-1, 1\}$$

More intuitively, each weak learner is a word associated with a class label. For example if we had a word `football`, and classes {`sports`,`non-sports`}, then we will have two weak learners from this word, namely

- *Predict* **sports** *if document has word* **football**
- *Predict* **non-sports** *if document has word* **football**.

1. [2 points] How many weak learners are there ?

★ **SOLUTION:** Two weak learners for each word, i.e. 2m weak learners.

2. This boosting algorithm can be used for feature selection. We run the algorithm and select the features in the *order in which they were identified* by the algorithm.

   (a) [4 points] Can this boosting algorithm select the same weak classifier more than once? Explain.

   ★ **SOLUTION:** The boosting algorithm optimizes each new α by assuming that all the previous votes remain fixed. It therefore does not optimize these coefficients jointly. The only way to correct the votes assigned to a weak learner later on is to introduce the same weak learner again. Since we only have a discrete set of possible weak learners here, it also makes sense to talk about selecting the exact same weak learner again.

   (b) [4 points] Consider ranking the features based on their individual mutual information with the class variable y, i.e. $\hat{I}(y; X_j)$. Will this ranking be more informative than the ranking returned by AdaBoost ? Explain.

   ★ **SOLUTION:** The boosting algorithm generates a linear combination of weak classifiers (here features). The algorithm therefore evaluates each new weak classifier (feature) relative to a linear prediction based on those already included. The mutual information criterion considers each feature individually and is therefore unable to recognize how multiple features might interact to benefit linear prediction.

4 [8 points] Gaussian Naive Bayes classifier

Consider the datasets **toydata1** in figure **3(A)** and **toydata2** in figure **3(B)**.

- In each of these datasets there are two classes, '+' and 'o'.
- Each class has the same number of points.
- Each data point has two real valued features, the X and Y coordinates.

For each of these datasets, draw the decision boundary that a Gaussian Naive Bayes classifier will learn.

★ **SOLUTION:** For **toydata1** the crucial detail is that GNB learns diagonal covariance matrices yielding axis aligned Gaussians. In figure 4(A) the two circles are the gaussians learned by GNB, and hence the decision surface is the tangent through the point of contact.

For **toydata2** GNB learns two Gaussians , one for the circle inside with small variance , and one for the circle outside with a much larger variance, and the decision surface is roughly shown in figure 4(B).

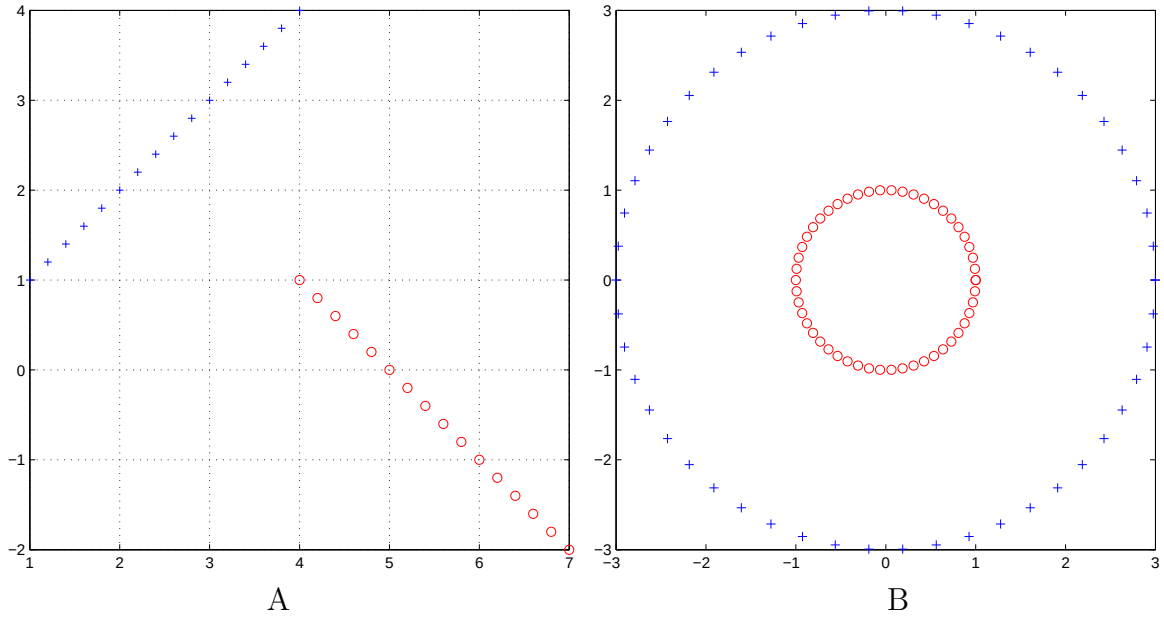


Figure 3: A. **toydata1** in Question 4, B. **toydata2** in Question 4

Remember that a very important piece of information was that all the classes had the *same* number of points, and so we don't have to worry about the prior.

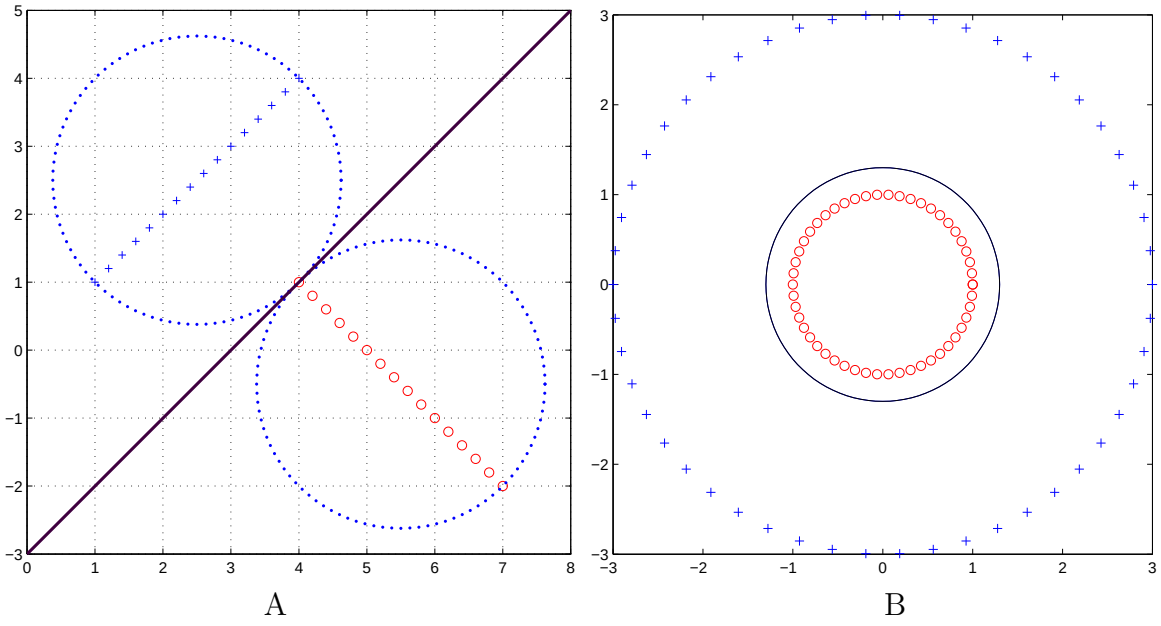
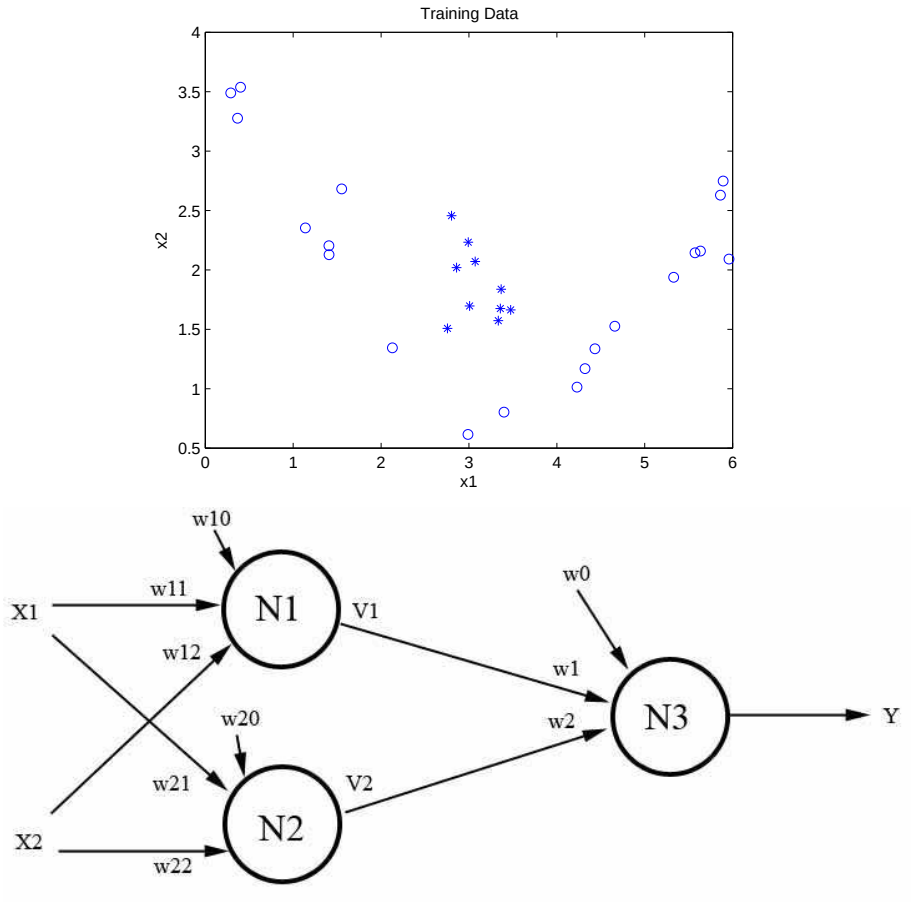


Figure 4: Solutions for A. **toydata1** in Question 4, B. **toydata2** in Question 4

5 [16 Points] Neural Networks

Consider the following classification training data (where "*" = true or 1 and "O" = false or 0) and neural network model that uses the **sigmoid** response function $\left(g(t) = \frac{1}{1+e^{-t}}\right)$.

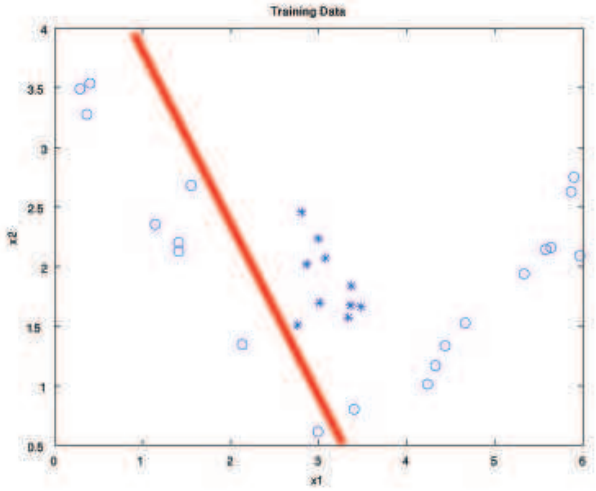


5.1 Weight choice

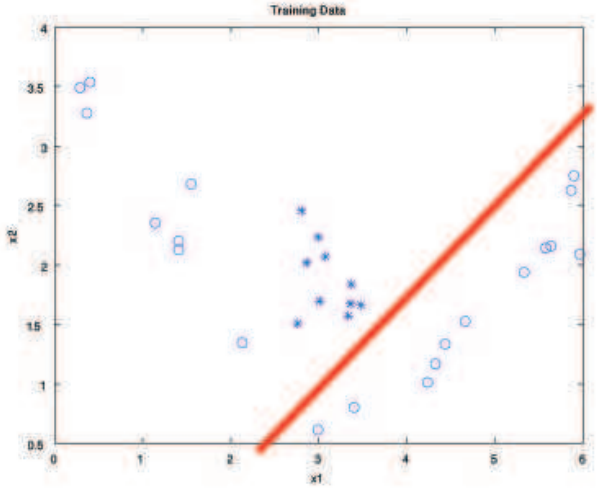
[8 points] We would like to set the weights (w) of the neural network so that it is capable of correctly classifying this dataset. Please plot the decision boundaries for N_1 and N_2 (e.g., for neuron N_1, the line where $w10 + w11 * X1 + w12 * X2 = 0$) on the first two graphs. In the third graph, which has axes V_2 and V_1, plot $\{V_1(x_1, x_2), V_2(x_1, x_2)\}$ for a few of the training points and provide a decision boundary so that the neural net will correctly classify the training data.

All graphs are on the following page!

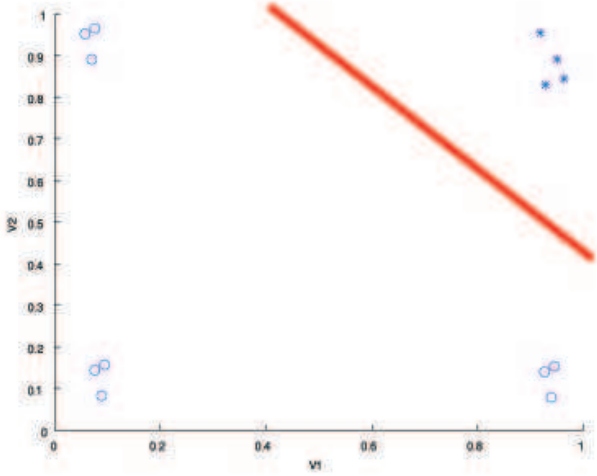
N1 (2 points)



N2 (2 points)



N3 (4 points)



5.2 Regularized Neural Networks

[8 points]

One method for preventing the neural networks' weights from overfitting is to add regularization terms. You will now derive the update rules for the regularized neural network. **Note:** $Y = out(x)$

Recall that the non-regularized gradient descent update rule for w_1 is:

$$w_1^{t+1} \leftarrow w_1^t + \eta \sum_j \left[\left(y^{(j)} - out(x^{(j)}) \right) out(x^{(j)})(1 - out(x^{(j)})) * V_1(x^{(j)}) \right] \tag{1}$$

[4 points] Derive the update rule for w_1 in the regularized neural net loss function which penalizes based on the square of each weight. Use λ to denote the magic regularization parameter.

★ **SOLUTION:** The regularization term is $\lambda(\sum_i w_i^2)$. Differentiating with respect to w_1 yields $2\lambda w_1$. The update rule is

$$w_1^{t+1} \leftarrow w_1^t + \eta(\sum_j \left[\left(y^{(j)} - out(x^{(j)}) \right) out(x^{(j)})(1 - out(x^{(j)})) * V_1(x^{(j)}) \right] - 2\lambda w_1)$$

[4 points] Now, re-express the regularized update rule so that the only difference between the regularized setting and the unregularized setting above is that the old weight w_1^t is scaled by some constant. Explain how this scaling prevents overfitting.

★ **SOLUTION:**

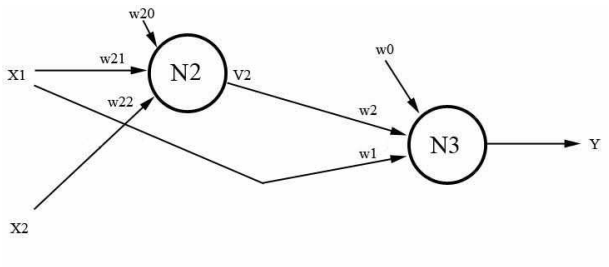
$$w_1^{t+1} \leftarrow w_1^t(1 - 2\eta\lambda) + \eta \sum_j \left[\left(y^{(j)} - out(x^{(j)}) \right) out(x^{(j)})(1 - out(x^{(j)})) * V_1(x^{(j)}) \right]$$

At each update the weight is kept closer to zero by the $(1 - 2\eta\lambda)$ term. This prevents the weights from becoming very large, which corresponds to overfitting.

5.3 Neural Net Simplification [Extra Credit (5 points)]

Please provide a feed-forward neural network with a smaller architecture (i.e., fewer neurons and weights) that is able to correctly predict the entire training set. Justify your answer.

★ **SOLUTION:** One solution follows from the observation that the decision boundary for N1 could be $x_1 = 3.7$. In fact, $N1$ can be removed entirely from the model. This yields a similar decision boundary for $N3$ except that $V1 = x_1$ ranges from 0 to 6.



Other possible solutions are to change the input feature space (e.g., by adding x_1^2 as an input to a single neuron along with x_1, x_2).

6 [14 Points] The Effect of Irrelevant Features

1. (a) [3 points] Provide a 2D dataset where 1-nearest neighbor (1-NN) has lower leave-one-out cross validation error (LOO error) than SVMs.

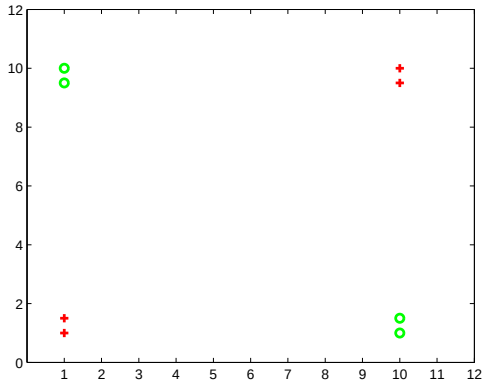


Figure 5: Dataset where 1-NN will do better than SVM in LOOCV.

(b) [3 points] Provide a 2D dataset where 1-NN has higher LOO error than SVMs.

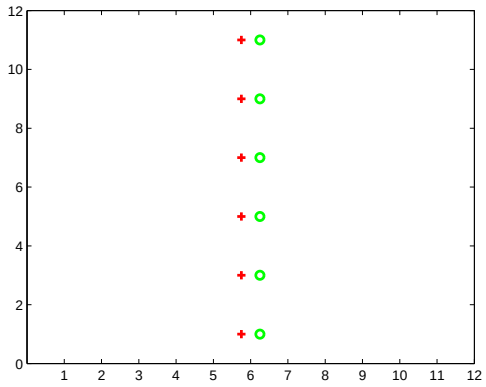


Figure 6: Dataset where SVM will do better than 1-NN in LOOCV.

2. [8 points] You will now generate a dataset to illustrate SVMs' robustness to irrelevant features. In particular, create a 2D dataset with features X_1 and X_2, here X_2 will be the irrelevant feature, such that:

   - If you only use X_1, 1-NN will have lower LOO error than SVMs,
   - but if you use both X_1 and X_2, the SVM LOO error will remain the same, but LOO error for 1-NN will increase significantly.

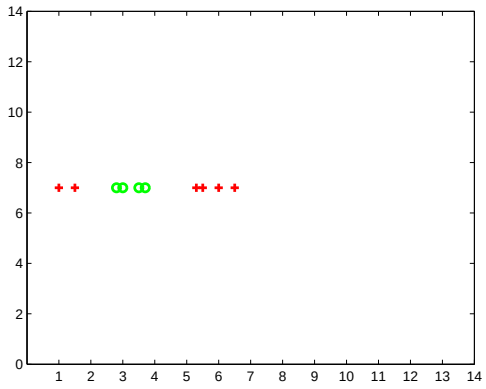


Figure 7: Here the horizontal axis is X_1, and we ignore X_2. 1-NN is perfect, and SVM will get the two points on the left wrong in LOOCV.

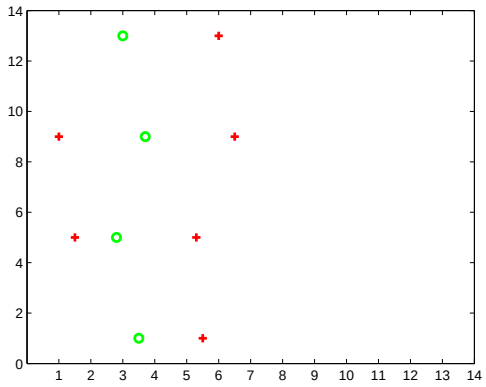


Figure 8: Here the horizontal axis is X_1, and the vertical axis is X_2. X_2 is the irrelevant feature. 1-NN gets every point wrong in LOOCV, while SVM has the same error.

You will receive extra credit if the 1-NN LOO error before adding the irrelevant feature is zero, but the error becomes 100% after adding the feature.

3. [Extra Credit (3 points)] SVMs tend to be robust to irrelevant features. Suppose we run SVMs with features $X_1, \ldots, X_n$, and then add a irrelevant feature X_{n+1} that cannot help increase the margin. How will SVMs automatically ignore this feature? Justify your answer formally.

★ **SOLUTION:** SVMs will automatically ignore this feature because it cannot possibly increase the margin, so giving it non-zero weight keeps the same margin but increases the regularization penalty. Therefore the solution with zero weight is superior to (i.e. has smaller objective function) all feasible solutions of the QP where this feature has non-zero weight.

7 [15 points] Learning Theory

Consider binary data-points X in n dimensions, with binary labels Y, i.e. $X \in \{0,1\}^n$; $Y \in \{0,1\}$. We wish to learn a mapping $X \to Y$ using a few different hypothesis classes, but are concerned about the tradeoff between the expressivity of our hypothesis space and the number of training examples required to learn the true mapping probably approximately correctly.

1. Consider the following hypothesis class H: decision stumps that choose a value for Y based on the value of one of the attributes of X. For example, there are two hypotheses in H that involve feature i:

$$h_i(X) = \begin{cases} 1 & \text{if } X_i = 1 \\ 0 & \text{otherwise} \end{cases} \qquad h_{\neg i}(X) = \begin{cases} 0 & \text{if } X_i = 1 \\ 1 & \text{otherwise;} \end{cases}$$

   (a) [3 points] What is the size of this hypothesis class?

   ★ **SOLUTION:** $H = 2n$

   (b) [3 points] For given ϵ, δ how many training examples are needed to yield a decision stump that satisfies the Haussler-PAC bound?

   ★ **SOLUTION:**

$$|H|e^{-m\epsilon} \leq \delta$$

$$\begin{aligned} m &\geq \frac{1}{\epsilon}\left(\log|H| + \log\frac{1}{\delta}\right) \\ &= \frac{1}{\epsilon}\left(\log(2n) + \log\frac{1}{\delta}\right) \end{aligned}$$

2. Now let us define another hypothesis class H', where each hypothesis is a majority over a set of simple decision stumps. Specifically, for each feature i, we either use h_i or $h_{\neg i}$, and the output is the result of a majority vote (in the case of a tie, we predict 1). For example, if we have 5 features, and we choose the stumps $\{h_{\neg 1}, h_2, h_3, h_4, h_{\neg 5}\}$, then the resulting hypothesis is:

$$h'(X) = \begin{cases} 1 & \text{if } h_{\neg 1}(X) + h_2(X) + h_3(X) + h_4(X) + h_{\neg 5}(X) \geq \frac{5}{2} \\ 0 & \text{otherwise} \end{cases}$$

(a) [4 points] What is the size of this hypothesis class?

★ **SOLUTION:** Each element of the hypothesis class here corresponds to picking a subset of n features. Hence:

$$|H| = 2^n$$

(b) [2 points] For given ϵ, δ how many training examples are needed to yield a hypothesis that satisfies the Haussler-PAC bound?

★ **SOLUTION:**

$$m \geq \frac{1}{\epsilon}\left(n \log 2 + \log \frac{1}{\delta}\right)$$

3. [3 points] What can we say about the amount of extra samples necessary to learn this voting classifier? Is this a concern?
Briefly explain the tradeoff between the expressive power of the hypothesis space and the number of training samples required for these two classifier.

★ **SOLUTION:** In the first part of the problem, the required number of samples scales as $O(\log n)$, and in the second part of the problem, it scales as $O(n)$. So we need more samples to get the PAC bound for the second hypothesis class, but scaling linearly in the number of features is probably acceptable.

The tradeoff illustrated here is that greater expressive power (as with H') necessitates more training samples. The smaller and less expressive class H requires fewer training examples.