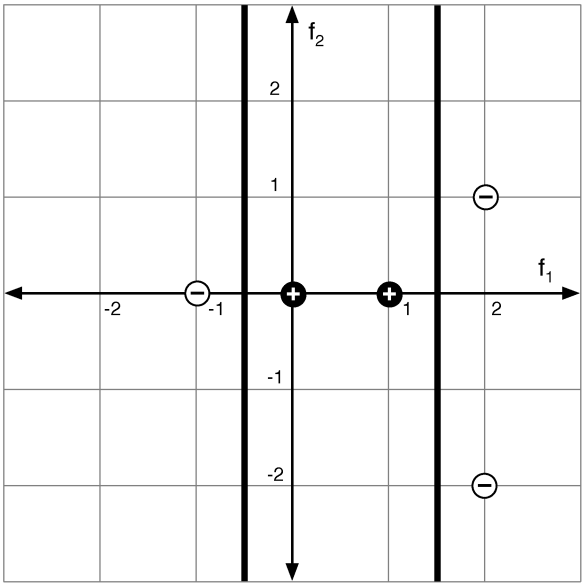
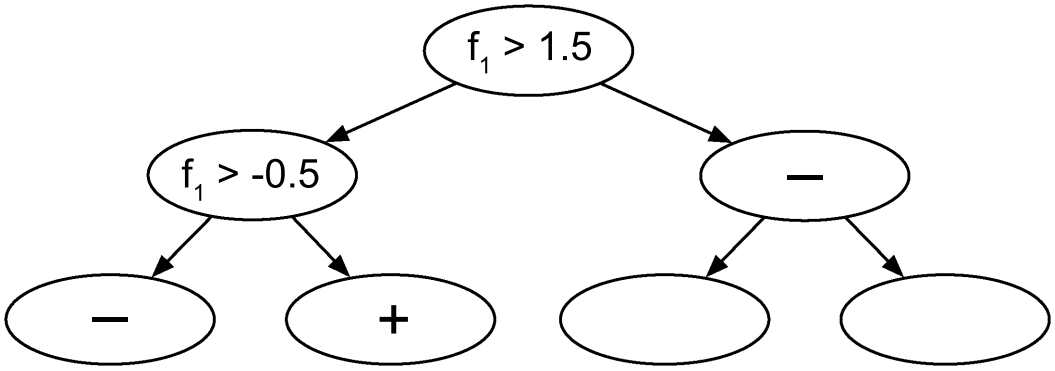
1 Decision Trees (13 pts)



Data points are: Negative: (-1, 0) (2, 1) (2, -2) Positive: (0, 0) (1, 0)

Construct a decision tree using the algorithm described in the notes for the data above.

1. Show the tree you constructed in the diagram below. The diagram is more than big enough, leave any parts that you don't need blank.



2. Draw the decision boundaries on the graph at the top of the page.

3. Explain how you chose the top-level test in the tree. The following table may be useful.

x	y	-(x/y)*lg(x/y)		x	y	-(x/y)*lg(x/y)
1	2	0.50		1	5	0.46
1	3	0.53		2	5	0.53
2	3	0.39		3	5	0.44
1	4	0.50		4	5	0.26
3	4	0.31				

Pick the decision boundary which falls halfway between each pair of adjacent points in each dimension, and which produces the minimum average entropy

$$\begin{aligned} AE &= q_< H(p_<) + (1 - q_<) H(p_>) \\ H(p) &= -p \lg(p) - (1-p) \lg(1-p) \end{aligned}$$

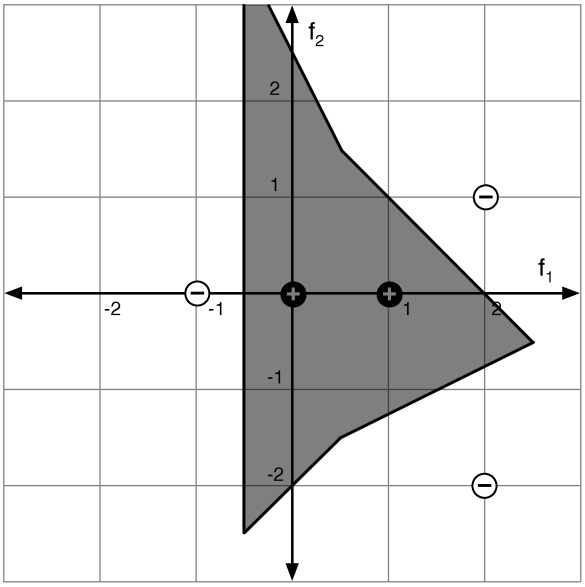
where $q_<$ is the fraction of points below the decision boundary, and $p_<$, $p_>$ are the fraction of positive (+) points below and above the decision boundary, respectively.

$$\begin{aligned} f_2 > \pm 0.5: &\quad AE = \frac{1}{5}(0) + \frac{4}{5}(1) = 0.8 \\ f_1 > 1.5: &\quad AE = \frac{3}{5} H\left(\frac{2}{3}\right) + \frac{2}{5}(0) = \frac{3}{5}(0.39 + 0.53) = 0.552 \\ f_1 > 0.5: &\quad AE = \frac{2}{5}(1) + \frac{3}{5} H\left(\frac{1}{3}\right) = \frac{2}{5} + \frac{3}{5}(0.39 + 0.53) = 0.952 \\ f_1 > -0.5: &\quad AE = \frac{1}{5}(0) + \frac{4}{5}(1) = 0.8 \end{aligned}$$

4. What class does the decision tree predict for the new point: (1, -1.01)

Positive (+)

2 Nearest Neighbors (8 pts)



Data points are: Negative: (-1, 0) (2, 1) (2, -2) Positive: (0, 0) (1, 0)

1. Draw the decision boundaries for 1-Nearest Neighbors on the graph above. Try to get the integer-valued coordinate points in the diagram on the correct side of the boundary lines.

2. What class does 1-NN predict for the new point: (1, -1.01) Explain why.

   Positive (+) since this is the class of the closest data point (1,0).

3. What class does 3-NN predict for the new point: (1, -1.01) Explain why.

   Positive (+) since it is the majority class of the three closest data points (0,0), (1,0) and (2,-2).

5 Naive Bayes (8 pts)

Consider a Naive Bayes problem with three features, $x_1 \ldots x_3$. Imagine that we have seen a total of 12 training examples, 6 positive (with $y = 1$) and 6 negative (with $y = 0$). Here is a table with some of the counts:

	$y = 0$	$y = 1$
$x_1 = 1$	6	6
$x_2 = 1$	0	0
$x_3 = 1$	2	4

1. Supply the following estimated probabilities. Use the Laplacian correction.

   - $\Pr(x_1 = 1|y = 0) = \frac{6+1}{6+2} = \frac{7}{8}$

   - $\Pr(x_2 = 1|y = 1) = \frac{0+1}{6+2} = \frac{1}{8}$

   - $\Pr(x_3 = 0|y = 0) = 1 - \frac{2+1}{6+2} = \frac{5}{8}$

2. Which feature plays the largest role in deciding the class of a new instance? Why?

   x_3**, because it has the biggest difference in the likelihood of being true for the two different classes. The other two features carry no information about the class.**

6 Learning algorithms (16 pts)

For each of the learning situations below, say what learning algorithm would be best to use, and why.

1. You have about 1 million training examples in a 6-dimensional feature space. You only expect to be asked to classify 100 test examples.

   Nearest Neighbors is a good choice. The dimensionality is low and so appropriate for KNN. For KNN, training is very fast and since there are few classifications, the fact that this will be slow does not matter. With 1 million training examples, neural net and SVM will be extremely expensive to train. Naive Bayes is plausible on computational grounds but likely to be less accurate than KNN.

2. You are going to develop a classifier to recommend which children should be assigned to special education classes in kindergarten. The classifier has to be justified to the board of education before it is implemented.

   A Decision Tree is a good choice since the resulting classifier will need to be understandable to humans.

3. You are working for Am*z*n as it tries to take over the retailing world. You are trying to predict whether customer X will like a particular book, as a function of the input which is a vector of 1 million bits specifying whether each of Am*z*n's other customers liked the book. You will train a classifier on a very large data set of books, where the inputs are everyone else's preferences for that book, and the output is customer X's preference for that book. The classifier will have to be updated frequently and efficiently as new data comes in.

   Naive Bayes is a good choice since it is fast to train and update. The dimensionality is high for Nearest Neighbors and Decision Trees. SVM's have to be re-trained from scratch if the data changes. Neural Nets could be trained incrementally but it will generally take a lot of iterations to change the current settings of the weights.

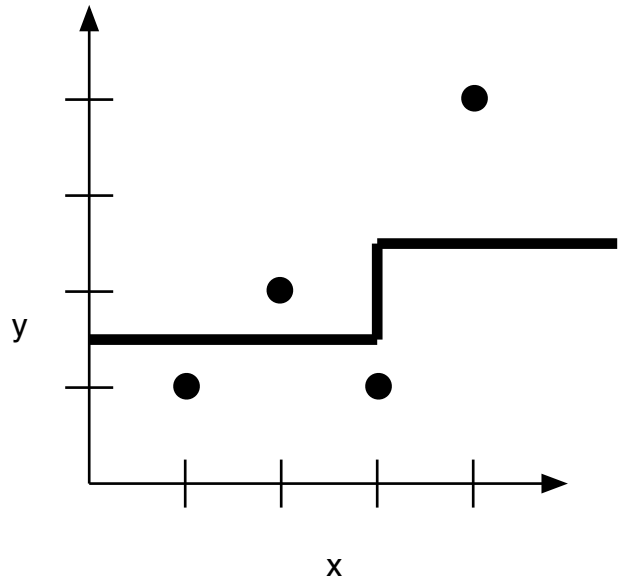
4. You are trying to predict the average rainfall in California as a function of the measured currents and tides in the Pacific ocean in the previous six months.

   This is a regression problem; neural nets with linear output functions, regression trees or locally weighted nearest neighbors are all appropriate choices.

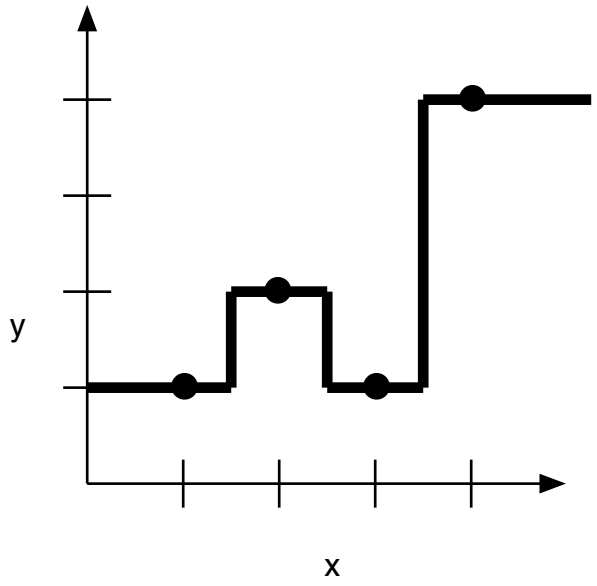
8 Regression (12 pts)

Consider a one-dimensional regression problem (predict y as a function of x). For each of the algorithms below, draw the approximate shape of the output of the algorithm, given the data points shown in the graph.

1. 2-nearest-neighbor (equally weighted averaging)



2. regression trees (with leaf size 1)

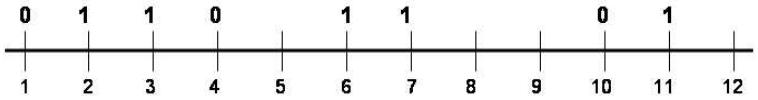


4 Machine Learning — Continuous Features (20 points)

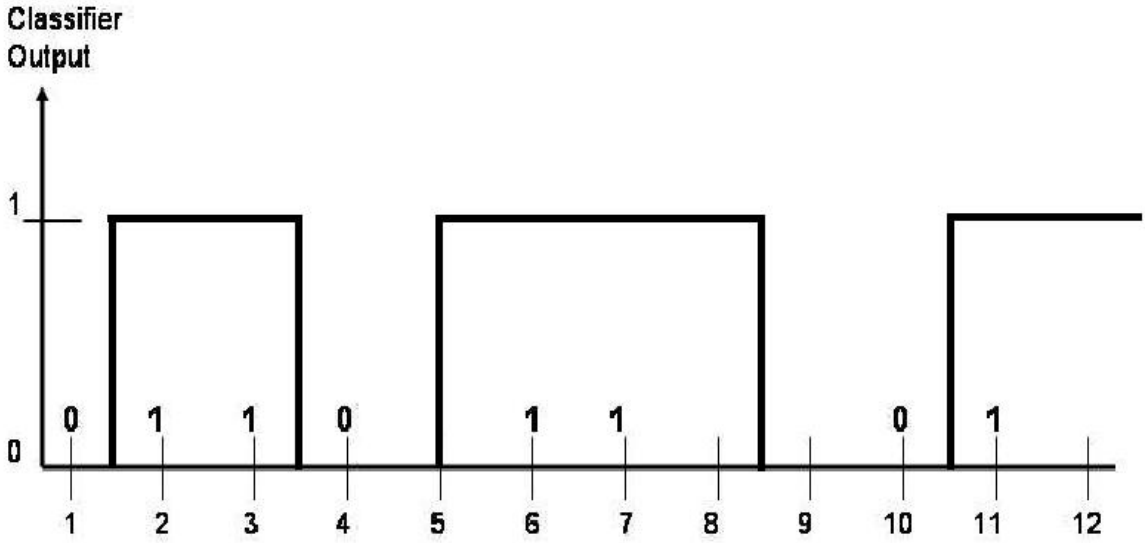
In all the parts of this problem we will be dealing with one-dimensional data, that is, a set of points (x^i) with only one feature (called simply x). The points are in two classes given by the value of y^i. We will show you the points on the x axis, labeled by their class values; we also give you a table of values.

4.1 Nearest Neighbors

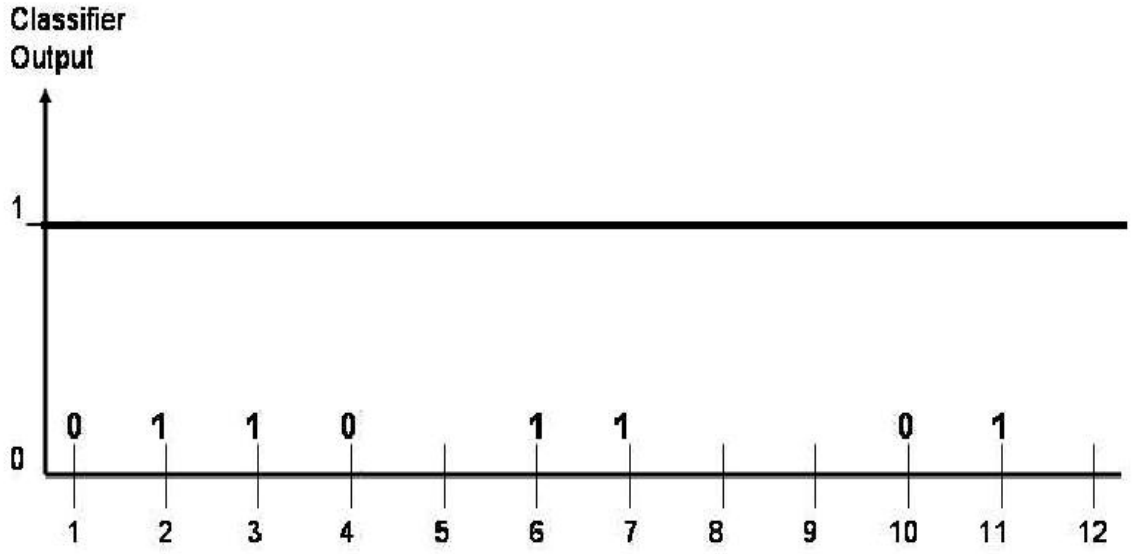
i	x^i	y^i
1	1	0
2	2	1
3	3	1
4	4	0
5	6	1
6	7	1
7	10	0
8	11	1



1. In the figure below, draw the output of a 1-Nearest-Neighbor classifier over the range indicated in the figure.

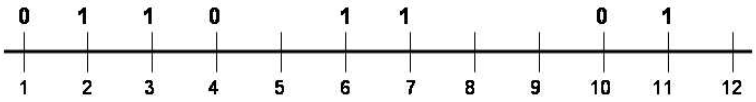


2. In the figure below, draw the output of a 5-Nearest-Neighbor classifier over the range indicated in the figure.



4.2 Decision Trees

Answer this problem using the same data as in the Nearest Neighbor problem above.



Which of the following three tests would be chosen as the top node in a decision tree?

$$x \leq 1.5 \qquad x \leq 5 \qquad x \leq 10.5$$

Justify your answer.

Recall that entropy for each side of a split is:

$$H = -p \log p - (1-p) \log(1-p)$$

So, for $x \leq 1.5$ *we have:*

$$\begin{aligned} H &= \frac{1(0) + 7\left(-\frac{5}{7}\log_2(\frac{5}{7}) - \frac{2}{7}\log_2(\frac{2}{7})\right)}{8} \\ H &= \frac{7(0.35 + 0.52)}{8} \\ H &= 0.761 \end{aligned}$$

while $x \leq 5$

$$\begin{aligned} H &= \frac{4\left(-\frac{2}{4}\log_2(\frac{2}{4}) - \frac{2}{4}\log_2(\frac{2}{4})\right) + 4\left(-\frac{3}{4}\log_2(\frac{3}{4}) - \frac{1}{4}\log_2(\frac{1}{4})\right)}{8} \\ H &= \frac{4(0.5 + 0.5) + 4(0.31 + 0.50)}{8} \\ H &= 0.905 \end{aligned}$$

and $x \leq 10.5$ *gives us:*

$$\begin{aligned} H &= \frac{1(0) + 7\left(-\frac{4}{7}\log_2(\frac{4}{7}) - \frac{3}{7}\log_2(\frac{3}{7})\right)}{8} \\ H &= \frac{7(0.46 + 0.52)}{8} \\ H &= 0.85 \end{aligned}$$

So, we choose the split with the least average entropy, which is $x \leq 1.5$.

6 Pruning Trees (20 points)

Following are some different strategies for pruning decision trees. We assume that we grow the decision tree until there is one or a small number of elements in each leaf. Then, we prune by deleting individual leaves of the tree until the score of the tree starts to get worse. The question is how to score each possible pruning of the tree.

For each possible definition of the score below, explain whether or not it would be a good idea and give a reason why or why not.

1. The score is the percentage correct of the tree on the training set.

   Not a good idea. The original tree was constructed to maximize performance on the training set. Pruning any part of the tree will reduce performance on the training set.

2. The score is the percentage correct of the tree on a separate validation set.

   A good idea. The validation set will be an independent check on whether pruning a node is likely to increase or decrease performance on unseen data.

3. The score is the percentage correct of the tree, computed using cross validation.

   Not a good idea. Cross-validation allows you to evaluate algorithms, not individual hypotheses. Cross-validation will construct many new hypotheses and average their performance, this will not tell you whether pruning a node in a particular hypothesis is worthwhile or not.

4. The score is the percentage correct of the tree, computed on the training set, minus a constant C times the number of nodes in the tree.

   C is chosen in advance by running this algorithm (grow a large tree then prune in order to maximize percent correct minus C times number of nodes) for many different values of C, and choosing the value of C that minimizes training-set error.

   Not a good idea. Running trials to maximize performance on the training set will not give us an indication of whether this algorithm will produce answers that generalize to other data sets.

5. The score is the percentage correct of the tree, computed on the training set, minus a constant C times the number of nodes in the tree.

   C is chosen in advance by running cross-validation trials of this algorithm (grow a large tree then prune in order to maximize percent correct minus C times number of nodes) for many different values of C, and choosing the value of C that minimizes cross-validation error.

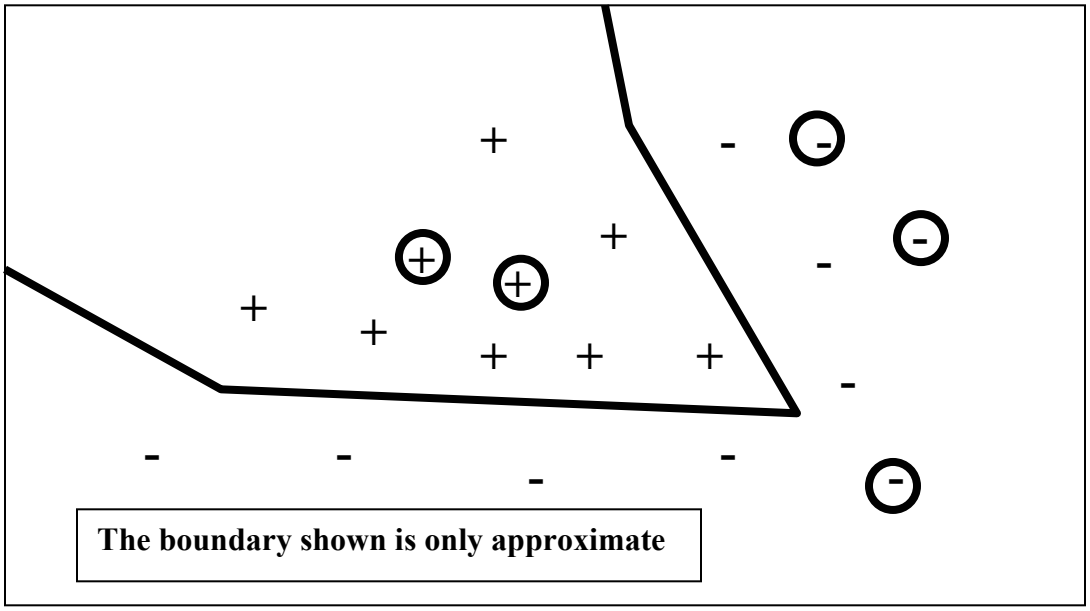
   A good idea when we don't have enough data to hold out a validation set. Choosing C by cross-validation will hopefully give us an effective general way of penalizing for complexity of the tree (for this type of data).

Problem 4: Learning (25 points)

Part A: (5 Points)

Since the cost of using a nearest neighbor classifier grows with the size of the training set, sometimes one tries to eliminate redundant points from the training set. These are points whose removal does not affect the behavior of the classifier for any possible new point.

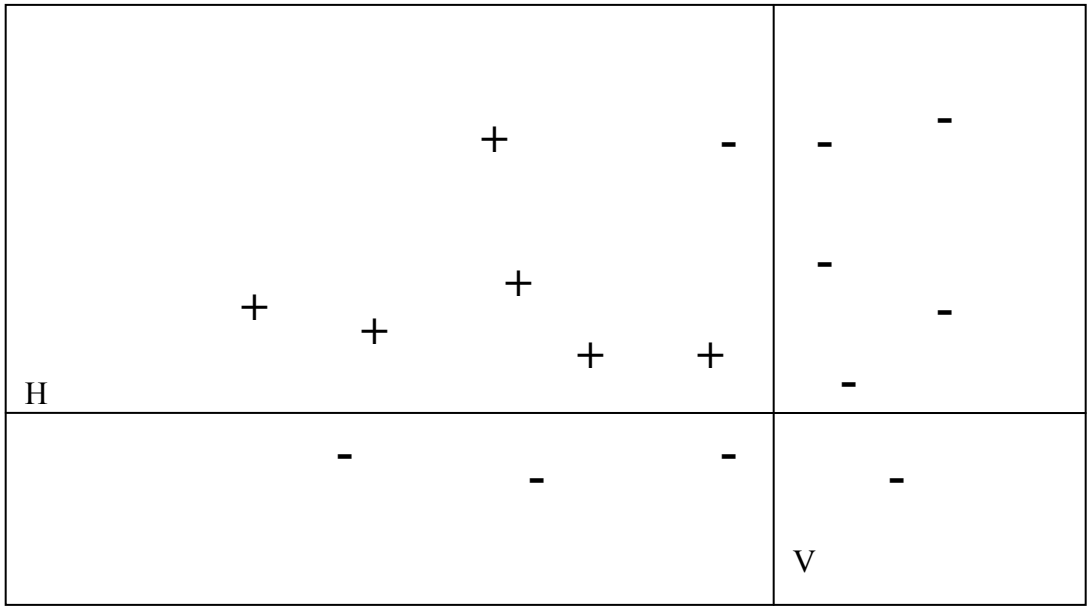
1. In the figure below, sketch the decision boundary for a 1-nearest-neighbor rule and circle the redundant points.



2. What is the general condition(s) required for a point to be declared redundant for a 1-nearest-neighor rule? Assume we have only two classes (+, -). Restating the definition of redundant ("removing it does not change anything") is not an acceptable answer. Hint – think about the neighborhood of redundant points.

Let the Voronoi cell for a training point be the set of points that are closest to that point (as opposed to some other training point). The Voronoi cell of a redundant point touches only on other Voronoi cells of points of the same class.

Part B: (5 Points)



Which of H or V would be preferred as an initial split for a decision (identification) tree? Justify your answer numerically.

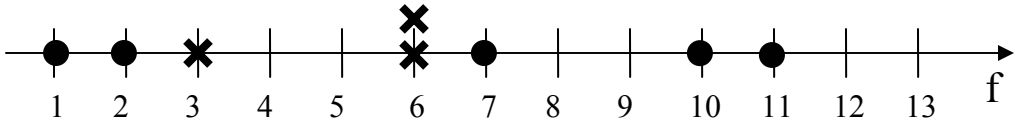
V = 0.60625
H = 0.75

So, V is chosen

x	y	-(x/y)*lg(x/y)
1	2	0.50
1	3	0.53
2	3	0.39
1	4	0.50
3	4	0.31
1	5	0.46
2	5	0.53
3	5	0.44
4	5	0.26
1	6	0.43
2	6	0.53
5	6	0.22
1	7	0.40
2	7	0.52
3	7	0.52
4	7	0.46
5	7	0.35
6	7	0.19

x	y	-(x/y)*lg(x/y)
1	8	0.38
3	8	0.53
5	8	0.42
7	8	0.17
1	9	0.35
2	9	0.48
4	9	0.52
5	9	0.47
7	9	0.28
8	9	0.15
1	10	0.33
3	10	0.52
7	10	0.36
9	10	0.14

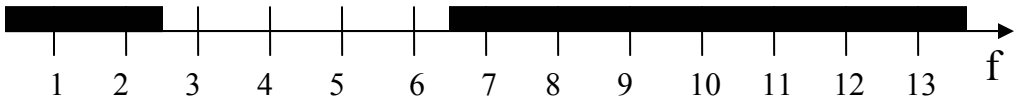
Problem 1: Classification (40 points)



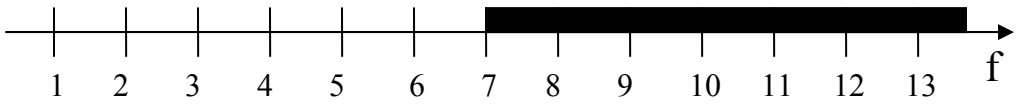
The picture above shows a data set with 8 data points, each with only one feature value, labeled f. Note that there are two data points with the same feature value of 6. These are shown as two X's one above the other, but they really should have been drawn as two X's on top of each other, since they have the same feature value.

Part A: (10 Points)

1. Consider using 1-Nearest Neighbors to classify unseen data points. On the line below, darken the segments of the line where the 1-NN rule would predict an O given the training data shown in the figure above.



2. Consider using 5-Nearest Neighbors to classify unseen data points. On the line below, darken the segments of the line where the 5-NN rule would predict an O given the training data shown in the figure above.

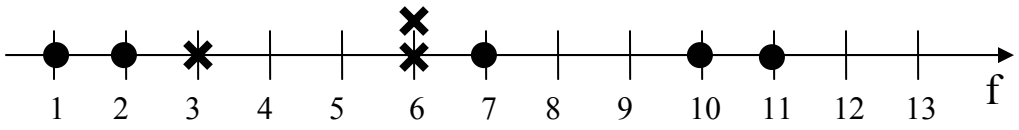


3. If we do 8-fold cross-validation using 1-NN on this data set, what would be the predicted performance? Settle ties by choosing the point on the left. Show how you arrived at your answer.

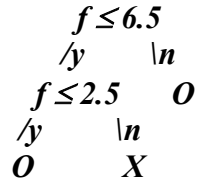
The point at 1 would be correct, nearest neighbor is at 2
The point at 2 would be correct, nearest neighbor is at 1 (tie)
The point at 3 would be incorrect, nearest neighbor is 2
Both points at 6 would be correct, nearest neighbor is the other point at 6.
The point at 7 would be incorrect, nearest neighbor is 6 (tie)
The point at 10 would be correct, nearest neighbor is 11
The point at 11 would be correct, nearest neighbor is 10

So,6 correct, 2 incorrect => 75% would be predicted performance

Part B: (8 Points)



Using this same data set, show the decision tree that would be built from this data. Assume that the tests in the tree are of the form f ≤ c. For each test show the approximate value of the average disorder for that test. To help you compute this, there's a small table of values of –(x/y)*log(x/y) for small integer x and y.



The top node as average disorder = 5/8[-2/5lg(2/5)-3/5lg(3/5)] + 3/8*0 = 0.61
The other decision node has 0 average disorder.

x	y	-(x/y)*lg(x/y)
1	2	0.50
1	3	0.53
2	3	0.39
1	4	0.50
3	4	0.31
1	5	0.46
2	5	0.53
3	5	0.44
4	5	0.26
1	6	0.43
2	6	0.53
5	6	0.22
1	7	0.40
2	7	0.52
3	7	0.52
4	7	0.46
5	7	0.35
6	7	0.19

x	y	-(x/y)*lg(x/y)
1	8	0.38
3	8	0.53
5	8	0.42
7	8	0.17
1	9	0.35
2	9	0.48
4	9	0.52
5	9	0.47
7	9	0.28
8	9	0.15
1	10	0.33
3	10	0.52
7	10	0.36
9	10	0.14

Problem 2: Overfitting (20 points)

For each of the supervised learning methods that we have studied, indicate how the method could overfit the training data (consider both your design choices as well as the training) and what you can do to minimize this possibility. There may be more than one mechanism for overfitting, make sure that you identify them all.

Part A: Nearest Neighbors (5 Points)

1. How does it overfit?
   Every point in dataset (including noise) defines its own decision boundary.
   The distance function can be chosen to do well on training set but less well on new data.

2. How can you reduce overfitting?
   Use k-NN for larger k
   Use cross-validation to choose k and the distance function

Part B: Decision Trees (5 Points)

1. How does it overfit?
   By adding new tests to the tree to correctly classify every data point in the training set.

2. How can you reduce overfitting?
   By pruning the resulting tree based on performance on a validation set.

Problem 3: Spaminator (10 points)

Suppose that you want to build a program that detects whether an incoming e-mail message is spam or not. You decide to attack this using machine learning. So, you collect a large number of training messages and label them as spam or not-spam. You further decide that you will use the presence of individual words in the body of the message as features. That is, you collect every word found in the training set and assign to each one an index, from 1 to N. Then, given a message, you construct a feature vector with N entries and write in each entry a number that indicates how many times the word appears in that message.

Part A: (6 Points)

If you had to choose between a Nearest Neighbor implementation or an Decision Tree implementation, which would you choose? Justify your answer briefly both in terms of expected accuracy and efficiency of operation. Indicate the strength and weaknesses of each approach.

Nearest Neighbors does not work well in high dimensions.
The biggest problem in using Nearest Neighbor would be choosing which of the features are relevant (which is related to choosing the distance metric). This is particularly severe in this application because of the huge numbers of probably irrelevant features (words).
The ID tree approach would spend initial effort in choosing which words were relevant to making the decision.
Nearest neighbor is also very expensive during classification for high dimensional feature vectors. ID trees would be much more efficient.
So, ID trees would be clearly better choice on all criteria.
However, Naïve Bayes would probably be better than either of them.

Part B: (4 Points)

Assume that you wanted to reduce the size of the feature vectors (during training and classification), for each of the approaches below indicate why it might be a good or bad idea.

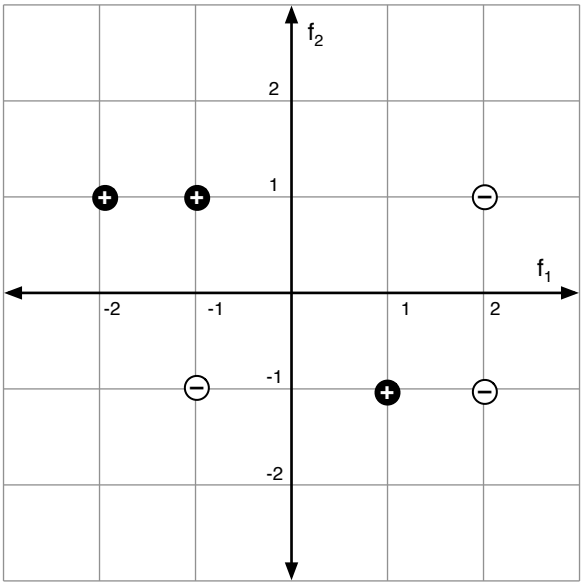
1. Use only the words that appear in spam messages.

   This would be a bad idea. There may be words that are common in spam and in non-spam. If you trained only with the spam words, you would end up deciding that many non-spam messages were spam.

2. Eliminate words that are very common in the whole data set.

This is generally a good idea. A count of "and", "is", "of", "the", etc. is not going to be very useful in differentiating spam from non-spam.

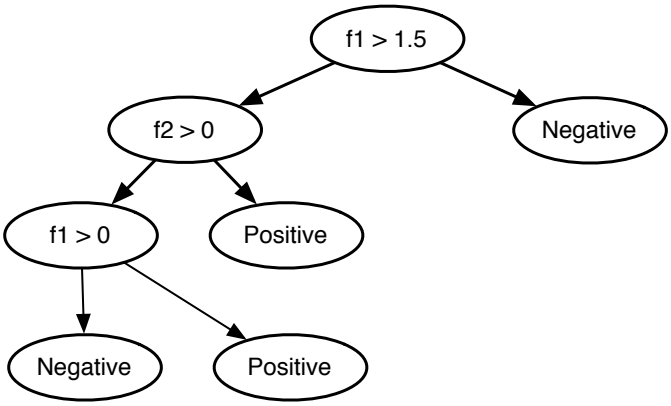
4 Decision Trees (20 points)



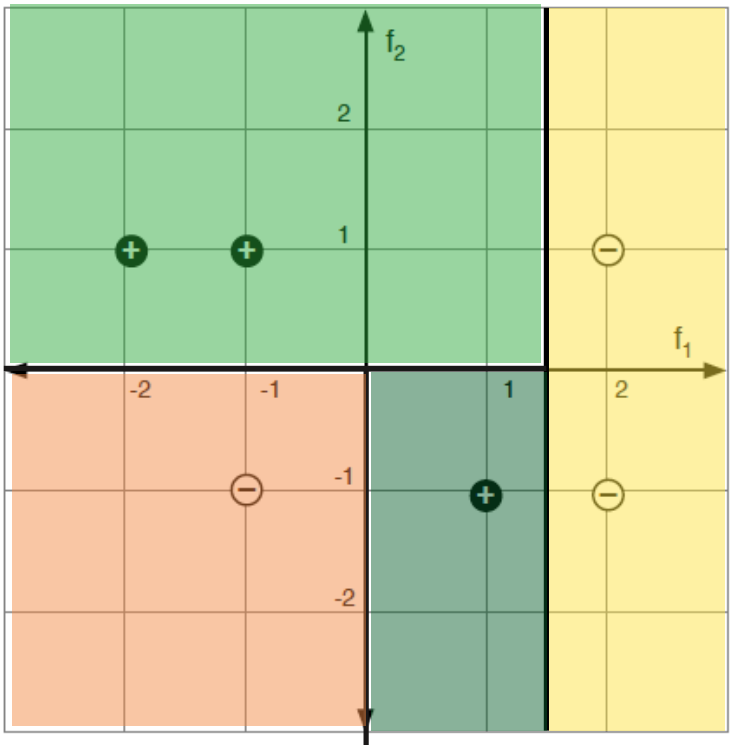
Data points are: Negative: (-1, -1) (2, 1) (2, -1) Positive: (-2, 1) (-1, 1) (1,-1)

Construct a decision tree using the algorithm described in the notes for the data above.

1. Show the tree you constructed in the diagram below. The diagram is more than big enough, leave any parts that you don't need blank. If you need to connect in the extra four nodes in the last row, add the connections to the parent nodes.



2. Draw the decision boundaries on the graph at the top of the page.



3. Explain how you chose the top-level test in the tree. The following table may be useful.

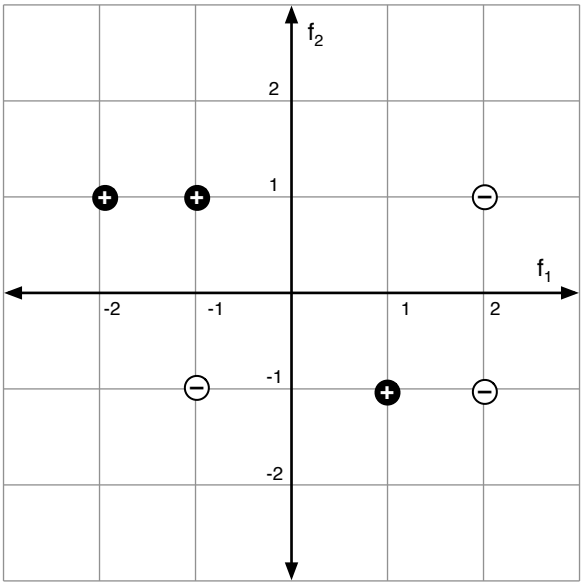
   The top level test (f1 ¿ 1.5) yields an entropy of
   $1/3 * (-(0log0) + (-2log2)) + 2/3 * (-(3/4)log(3/4) + (-1/4)log(1/4))$
   $= 2/3 * .81 = 0.54$

   Other cuts yield larger entropies:
   $f1 > -1.5 : 0.80$
   $f1 > 0 : 0.91$
   $f2 > 0 : .93$

4. What class does the decision tree predict for the new point: (1, 1)

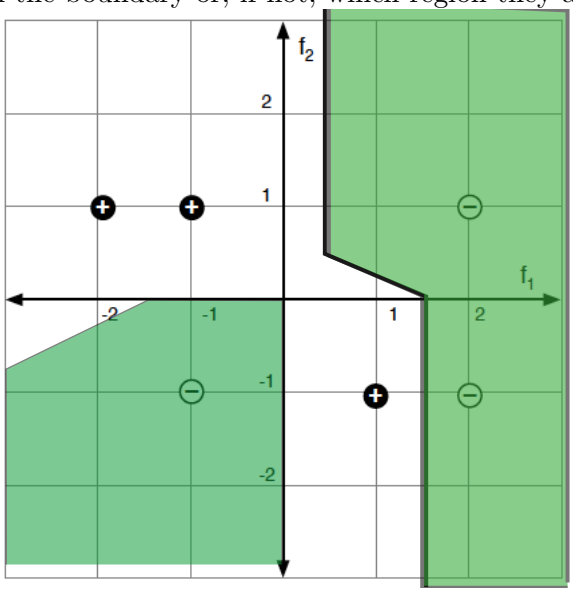
   Positive (+)

5 Nearest Neighbors (20 points)



Data points are: Negative: (-1, -1) (2, 1) (2, -1) Positive: (-2, 1) (-1, 1) (1,-1)

1. Draw the decision boundaries for 1-Nearest Neighbors on the graph above. Your drawing should be accurate enough so that we can tell whether the integer-valued coordinate points in the diagram are on the boundary or, if not, which region they are in.



2. What class does 1-NN predict for the new point: (1, 1) Explain why.
   Negative, it is closest to (2,1)

3. What class does 1-NN predict for the new point: (1, 0) Explain why.
   Positive, it is closest to (1,-1)

4. What class does 3-NN predict for the new point: (1, 0) Explain why.

   Negative, three closest points are (1,-1): Positive, (2,1): Negative and (2,-1) Negative.

5. In general, how would you select between two alternative values of k for use in k-nearest neighbors?

   Perform cross-validation using the two values and select the one with best average performance¿

6 Naive Bayes (15 points)

Consider a Naive Bayes problem with three features, $x_1 \ldots x_3$. Imagine that we have seen a total of 12 training examples, 6 positive (with $y = 1$) and 6 negative (with $y = 0$). Here are the actual points:

x_1	x_2	x_3	y
0	1	1	0
1	0	0	0
0	1	1	0
1	1	0	0
0	0	1	0
1	0	0	0
1	0	1	1
0	1	0	1
1	1	1	1
0	0	0	1
0	1	0	1
1	0	1	1

Here is a table with the summary counts:

	$y = 0$	$y = 1$
$x_1 = 1$	3	3
$x_2 = 1$	3	3
$x_3 = 1$	3	3

1. What are the values of the parameters $R_i(1, 0)$ and $R_i(1, 1)$ for each of the features i (using the Laplace correction)?

   All the parameters are $(3 + 1)/(6 + 2) = 0.5$

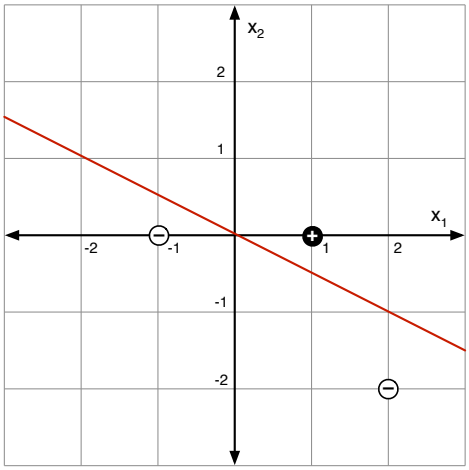
2. If you see the data point $1, 1, 1$ and use the parameters you found above, what output would Naive Bayes predict? Explain how you got the result.

   The prediction is arbitrary since S(0)=S(1) = 1/8

3. Naive Bayes doesn't work very well on this data, explain why.

   The basic assumption of NB is that the features are independent, give the class. In this data set, features 1 and 3 are definitely not independent; the values of these features are opposite for class 0 and equal for class 1. All the information is in these correlations, each feature independently says nothing about the class, so NB is not really applicable. Note that a decision tree would not have any problem with this data set.

3 Perceptron (7 pts)



Data points are: Negative: (-1, 0) (2, -2) Positive: (1, 0). Assume that the points are examined in the order given here. Recall that the perceptron algorithm uses the extended form of the data points in which a 1 is added as the 0th component.

1. The linear separator obtained by the standard perceptron algorithm (using a step size of 1.0 and a zero initial weight vector) is (0 1 2). Explain how this result was obtained.

   The perceptron algorithm cycles through the augmented points, updating weights according to the update rule $w_{\text{new}} = w + y \cdot x$ after misclassifying points. The intermediate weights are given in the table below.

Test point	misclassified?	Updated weights
Initial weights		**0 0 0**
-: (1 -1 0)	**yes**	**-1 1 0**
-: (1 2 -2)	**yes**	**-2 -1 2**
+: (1 1 0)	**yes**	**-1 0 2**
-: (1 -1 0)	**no**	
-: (1 2 -2)	**no**	
+: (1 1 0)	**yes**	**0 1 2**
-: (1 -1 0)	**no**	
-: (1 2 -2)	**no**	
+: (1 1 0)	**no**	

2. What class does this linear classifier predict for the new point: (2.0, -1.01)

   The margin of the point is -0.01, so it would be classified as negative.

3. Imagine we apply the perceptron learning algorithm to the 5 point data set we used on Problem 1: Negative: (-1, 0) (2, 1) (2, -2), Positive: (0, 0) (1, 0). Describe qualitatively what the result would be.

   The perceptron algorithm would not converge since the 5 point data set is not linearly separable.

7 Error versus complexity (15 pts)

Most learning algorithms we have seen try to find a hypotheses that minimizes error. But how do they attempt to control complexity? Here are some possible approaches:

A: Use a fixed-complexity hypothesis class

B: Include a complexity penalty in the measure of error

C: Nothing

For each of the following algorithms, specify which approach it uses and say what hypothesis class it uses (including any restrictions) and what complexity criterion (if any) is included in the measure of error. If the algorithm attempts to optimize the error measure, say whether it is guaranteed to find an optimal solution or just an approximation.

1. perceptron

   A. It uses a fixed hypothesis class of linear separators. It is guaranteed to find a separator if one exists.

2. linear SVM

   B. It includes a complexity penalty in the error criterion (which is to maximize the margin while separating the data). It optimizes this criterion.

3. decision tree with fixed depth

   A. It uses a fixed hypothesis class, which is the class of fixed-depth trees. Implicitly, it tries to find the lowest-error tree within this class, but isn't guaranteed to optimize that criterion.

4. neural network (no weight decay or early stopping)

   A. It uses a fixed hypothesis class, which is determined by the wiring diagram of the network.

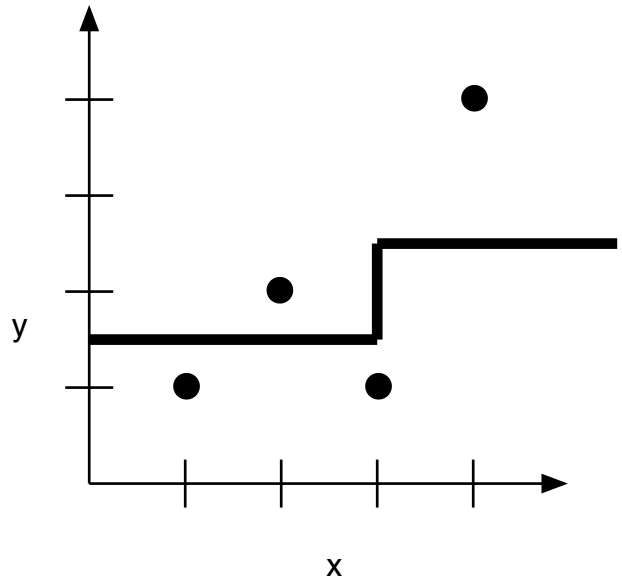
5. SVM (with arbitrary data and $c < \infty$)

   B. It includes a complexity penalty in the error criterion (which is to maximize the margin subject to assigning an $\alpha < c$ to each data point.

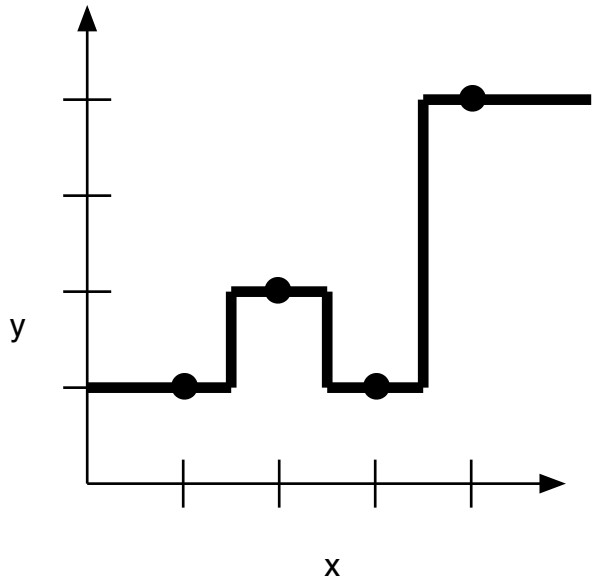
8 Regression (12 pts)

Consider a one-dimensional regression problem (predict y as a function of x). For each of the algorithms below, draw the approximate shape of the output of the algorithm, given the data points shown in the graph.

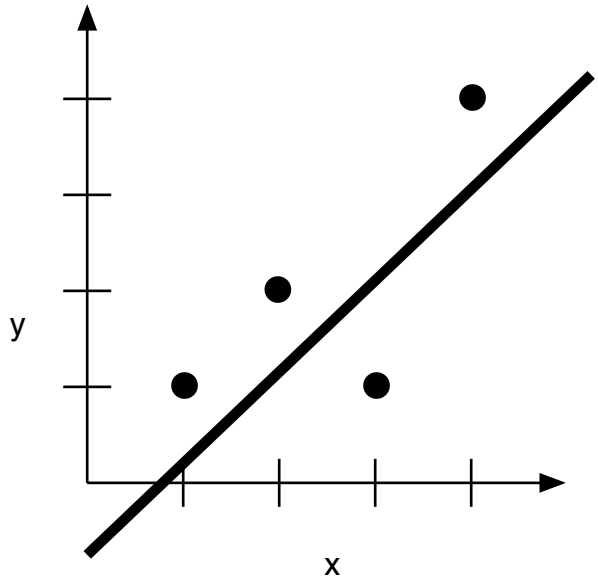
1. 2-nearest-neighbor (equally weighted averaging)



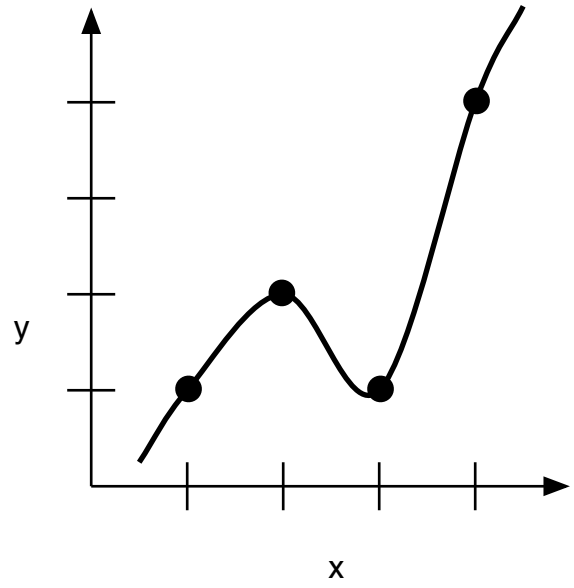
2. regression trees (with leaf size 1)



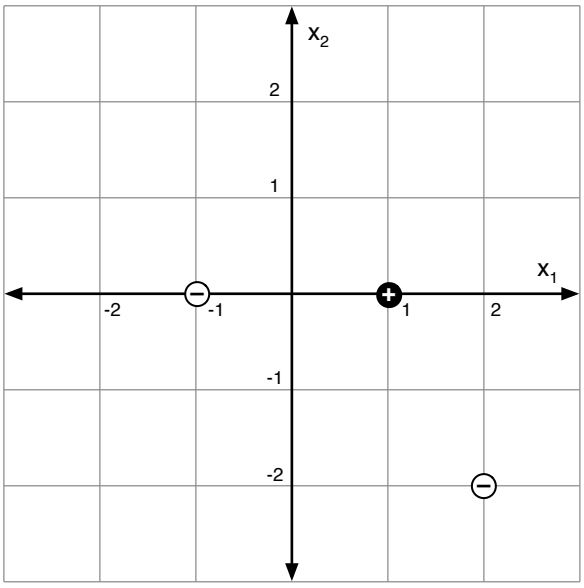
3. one linear neural-network unit



4. multi-layer neural network (with linear output unit)



9 SVM



Data points are: Negative: (-1, 0) (2, -2) Positive: (1, 0)

Recall that for SVMs, the negative class is represented by a desired output of -1 and the positive class by a desired output of 1.

1. For each of the following separators (for the data shown above), indicate whether they satisfy all the conditions required for a support vector machine, assuming a linear kernel. Justify your answers very briefly.

   (a) $x_1 + x_2 = 0$

   Goes through the (2,-2) point so obviously not maximal margin.

   (b) $x_1 + 1.5x_2 = 0$

   Yes. All three points are support vectors, with margin = 1.

   (c) $x_1 + 2x_2 = 0$

   No. The margin for (2,-2) is 2, not 1.

   (d) $2x_1 + 3x_2 = 0$

   No. The margin for the points is 2, not 1

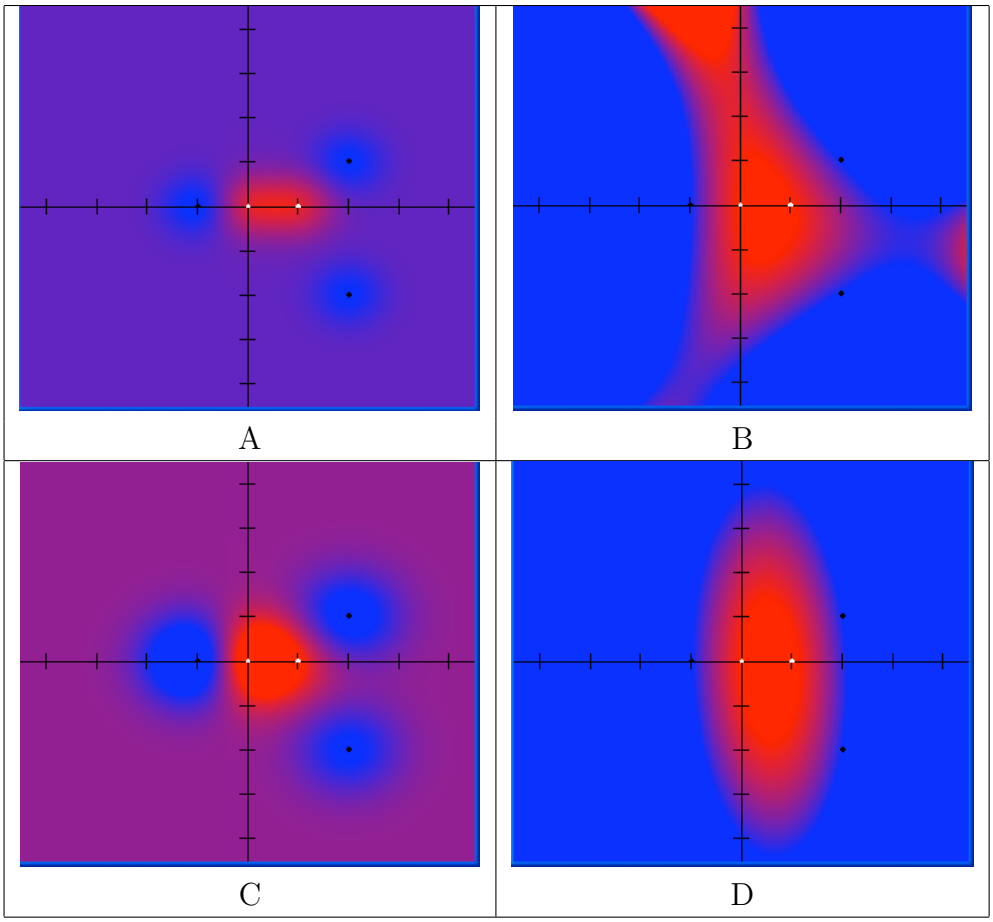
2. For each of the kernel choices below, find the decision boundary diagram (on the next page) that best matches. In these diagrams, the brightness of a point represents the magnitude of the SVM output; red means positive output and blue means negative. The black circles are the negative training points and the white circles are the positive training points.

   (a) Polynomial kernel, degree 2 : **D**

   (b) Polynomial kernel, degree 3 : **B**

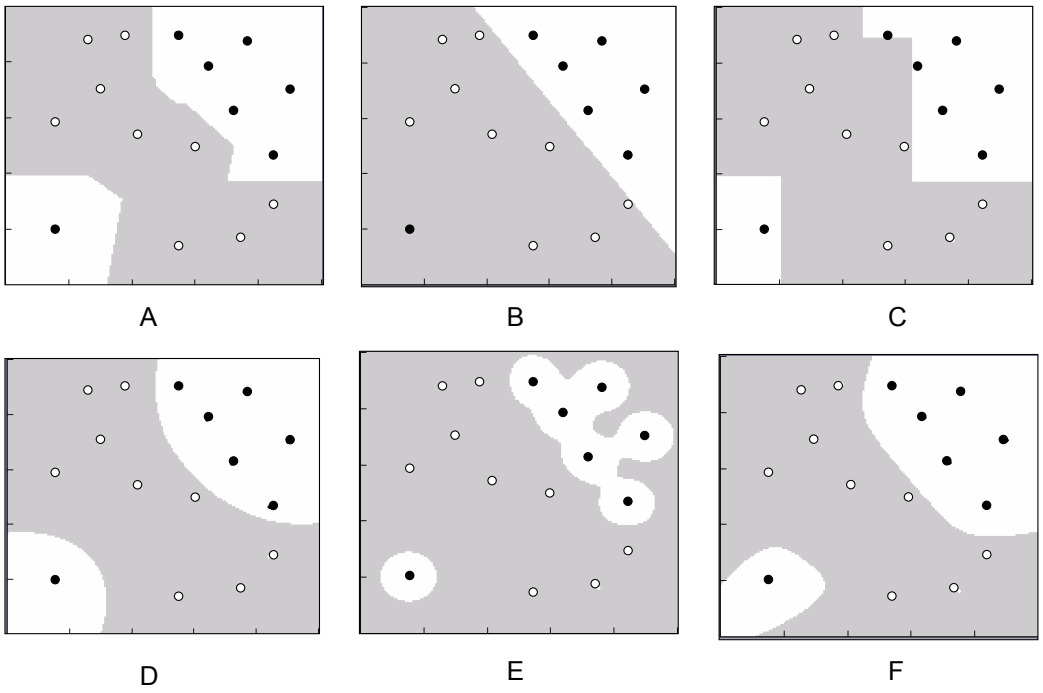
   (c) Radial basis kernel, sigma = 0.5 : **A**

   (d) Radial basis kernel, sigma = 1.0 : **C**


A
B
C
D

5 Learning hypothesis classes (16 points)

Consider a classification problem with two real-valued inputs. For each of the following algorithms, specify all of the separators below that it could have generated and explain why. If it could not have generated any of the separators, explain why not.



1. 1-nearest neighbor

   A — this is a Voronoi partition of the space, which describes the decision boundary produced by the 1-nearest neighbor algorithm

2. decision trees on real-valued inputs

   C — decision trees on real-valued inputs create a decision boundary that is made up of rectangles in the input space

3. standard perceptron algorithm

   none — the inputs are not linearly separable, and the standard perceptron algorithm does not terminate until it finds a linear separator that correctly classifies all of the training data

4. SVM with linear kernel

   B — the SVM algorithm will find some separator in the space that maximizes the margin, even if the data are not linearly separable

5. SVM with Gaussian kernel ($\sigma = 0.25$)

   E — a small sigma results in a classifier that more tightly fits the training data because the Gaussian bumps at each point are narrower

6. SVM with Gaussian kernel ($\sigma = 1$)

   D (or F) — a larger sigma results in a classifier that generalizes better because the Gaussian bumps at each point are wider. D is the separator actually generated by an SVM with Gaussian kernel, $\sigma = 1$, but we accepted F because it is difficult to tell which of these two would be generated without actually running the algorithm.

7. neural network with no hidden units and one sigmoidal output unit, run until convergence of training error

   B — a neural network with a single sigmoidal output will generate a linear classifier. The difference between a neural net with a single sigmoidal output and a perceptron unit is that the neural net training algorithm terminates when the error on a validation set reaches a minimum.

8. neural network with 4 hidden units and one sigmoidal output unit, run until convergence of training error

   F (and/or D) — a neural net with hidden units, run until convergence, can correctly classify all of the data (ruling out B). Also, the decision boundary will be smooth (why?) (ruling out A and C). Why not E?

6 Perceptron (8 points)

The following table shows a data set and the number of times each point is misclassified during a run of the perceptron algorithm, starting with zero weights. What is the equation of the separating line found by the algorithm, as a function of x_1, x_2, and x_3? Assume that the learning rate is 1 and the initial weights are all zero.

x_1	x_2	x_3	y	times misclassified
2	3	1	+1	12
2	4	0	+1	0
3	1	1	-1	3
1	1	0	-1	6
1	2	1	-1	11

$$\begin{aligned}\bar{w} &= \eta \sum_{i=1}^{m} \alpha_i y^i \bar{x}^i \\ &= (12)(1)(1,2,3,1) + (3)(-1)(1,3,1,1) + (6)(-1)(1,1,1,0) + (11)(-1)(1,1,2,1) \\ &= (-8,-2,5,-2)\end{aligned}$$

So the equation of the separating line is

$$-2x_1 + 5x_2 - 2x_3 - 8 = 0$$

7 SVMs (12 points)

Assume that we are using an SVM with a **polynomial kernel of degree 2**. You are given the following support vectors:

x_1	x_2	y
-1	2	+1
1	2	-1

The α values for each of these support vectors are equal to 0.05.

1. What is the value of b? Explain your approach to getting the answer.
   Answer: 0
2. What value does this SVM compute for the input point $(1, 3)$
   Answer: 0.05(1+(1,3).(-1,2))$^2 - 0.05(1 + (1, 3).(1, 2))^2] = 0.05[36 - 64] = -1.4$

4.4 SVM

What are the values for the α_i and the offset b that would give the maximal margin linear classifier for the two data points shown below? You should be able to find the answer without deriving it from the dual Lagrangian.

i	x^i	y^i
1	0	1
2	4	-1

We know that the $w = \sum_i \alpha_i x^i y^i$. *Thus:*

$$\begin{aligned} w &= \alpha_1 x^1 y^1 + \alpha_2 x^2 y^2 \\ w &= \alpha_1(0)(1) + \alpha_2(4)(-1) \\ w &= -4\alpha_2 \end{aligned}$$

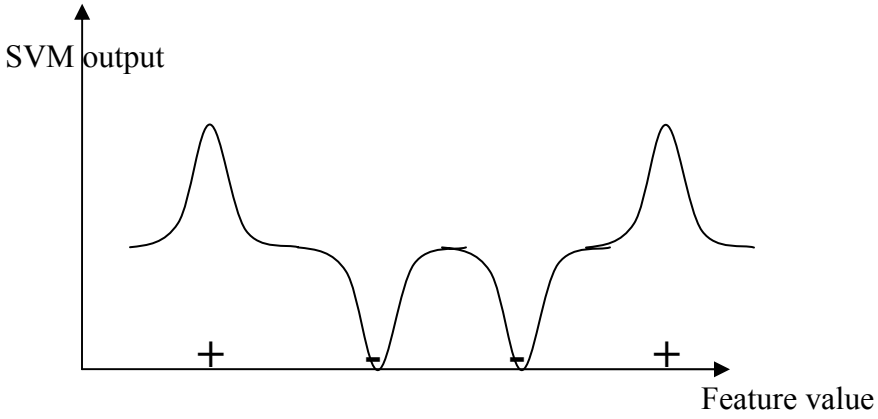
We know further that $\sum_i y^i \alpha_i = 0$, *so the alphas must be equal. Lastly, we know that the margin for the support vectors is 1, so* $wx_1 + b = 1$, *which tells us that* $b = 1$, *and* $wx_2 + b = -1$, *which tells us that* $w = -0.5$. *Thus we know that* $\alpha_1 = \alpha_2 = \frac{1}{8}$.

Part D: (10 Points)

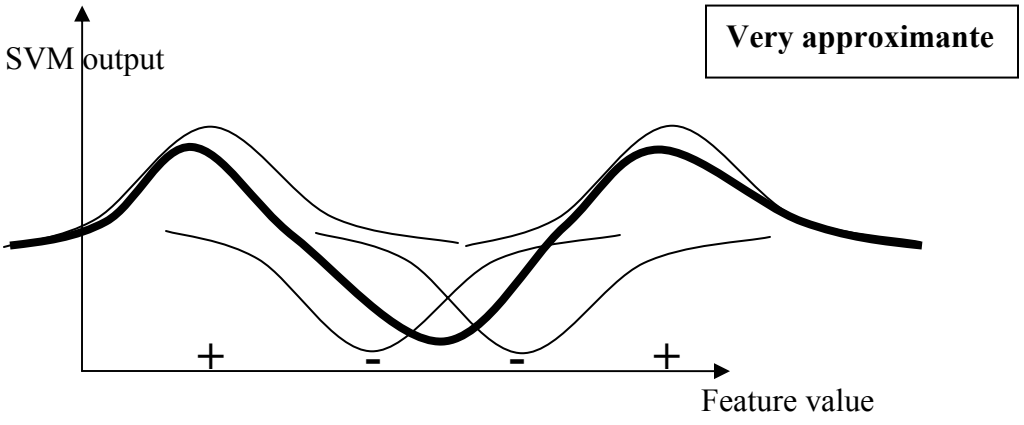
1. Consider the simple one-dimensional classification problem shown below. Imagine attacking this problem with an SVM using a radial-basis function kernel. Assume that we want the classifier to return a positive output for the + points and a negative output for the – points.

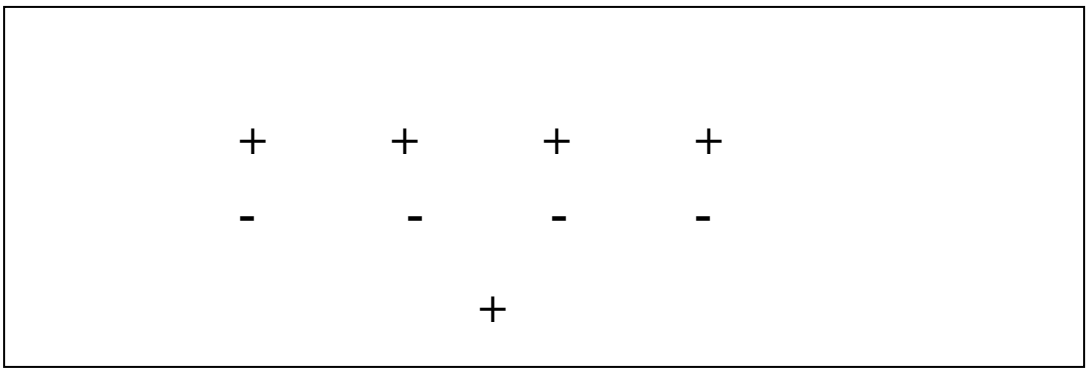
   Draw a plausible classifier output curve for a trained SVM, indicating the classifier output for every feature value in the range shown. Do this twice, once assuming that the standard deviation (σ) is very small relative to the distance between adjacent training points and again assuming that the standard deviation (σ) is about double the distance between adjacent training points.

   Small standard deviation (σ):



   Large standard deviation (σ):





2. Would you expect that a polynomial kernel with d=1 would be successful in carrying out the classification shown above? Explain.

   No, a first degree polynomial is a line and this is not linearly separable.

3. Assume we use an SVM with a radial-basis function kernel to classify these same data points. We repeat the classification for different values of the the standard deviation (σ) used in the kernel. What would you expect to be the relationship between the standard deviation used in the kernel and the value of the largest Lagrange multiplier (a_i) needed to carry out the classification? That is, would you expect that the max a_i would increase or decrease as the standard deviation decreases? Explain your answer.

   As the standard deviation decreases, we expect the max Lagrange multiplier to decrease. When the Gaussians overlap a lot, the multiplier for the isolated + has to be quite big to "raise" the classifier output to positive. When the Gaussians don't overlap, the multiplier will not need to be large.

Part E: (5 Points)

Given a validation set (a set of samples which is separate from the training set), explain how it should be used in connection with training different learning functions (be specific about the problems that are being addressed):

1. For a neural net

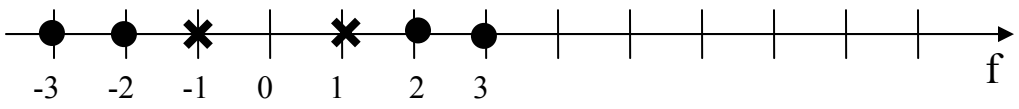
Use the validation set to:

   a. *choose the number of units in the network*
   b. *decide when to stop backpropagation*

2. For a decision (identification) tree

Use the validation set for pruning the tree – that is, drop tests that do not improve the performance on validation set.

Part D: (10 Points)



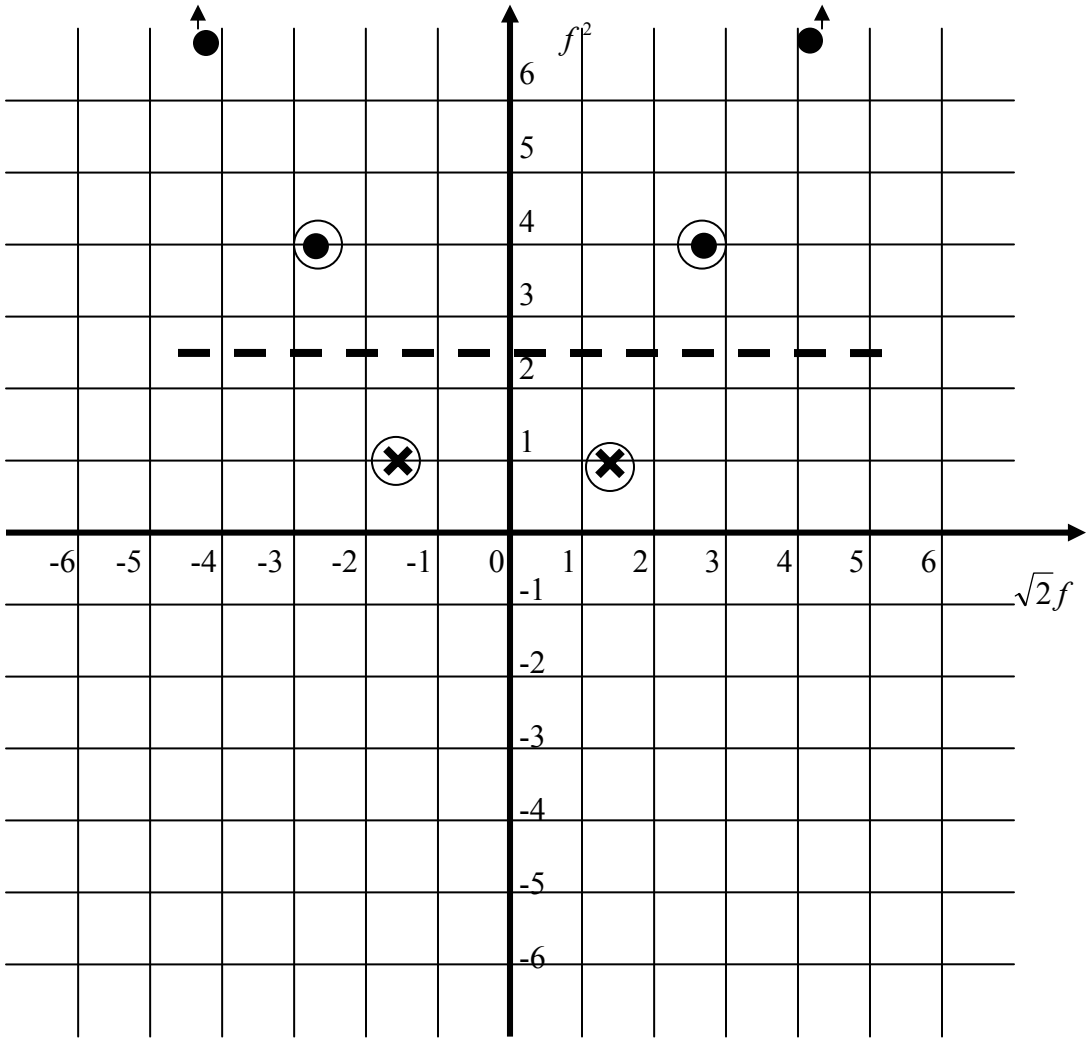
Consider the simplified data set above and consider using an SVM with a polynomial kernel with d=2. Let's say the data points are specified as:

X1 = [-2] Y1= -1
X2 = [-1] Y2= 1
X3 = [1] Y3= 1
X4 = [2] Y4= -1
X5=[-3] Y5= -1
X6=[3] Y6= -1

1. What are the kernel values?

K(x1,x1)	***25***
K(x1,x2)	***9***
K(x2,x3)	***0***
K(x3,x4)	***9***

2. Show a reasonably accurate picture of the **transformed** feature space.
   a. label the axes,
   b. label the data points,
   c. show the separating line that would be found by the SVM,
   d. circle the support vectors.



Problem 2: Overfitting (20 points)

For each of the supervised learning methods that we have studied, indicate how the method could overfit the training data (consider both your design choices as well as the training) and what you can do to minimize this possibility. There may be more than one mechanism for overfitting, make sure that you identify them all.

Part A: Nearest Neighbors (5 Points)

1. How does it overfit?
   Every point in dataset (including noise) defines its own decision boundary.
   The distance function can be chosen to do well on training set but less well on new data.

2. How can you reduce overfitting?
   Use k-NN for larger k
   Use cross-validation to choose k and the distance function

Part B: Decision Trees (5 Points)

1. How does it overfit?
   By adding new tests to the tree to correctly classify every data point in the training set.

2. How can you reduce overfitting?
   By pruning the resulting tree based on performance on a validation set.

Part C: Neural Nets (5 Points)

1. How does it overfit?

   By having too many units and therefore too many weights, thus enabling it to fit every nuance of the training set.

   By training too long so as to fit the training data better.

2. How can you reduce overfitting?

   Using cross-validation to choose a not too complex network

   By using a validation set to decide when to stop training.

Part D: SVM [Radial Basis and Polynomial kernels] (5 Points)

1. How does it overfit?

   In RBF, by choosing a value of sigma (the std dev of Gaussian) too small.

   In Polynomial, by choosing the degree of the polynomial too high

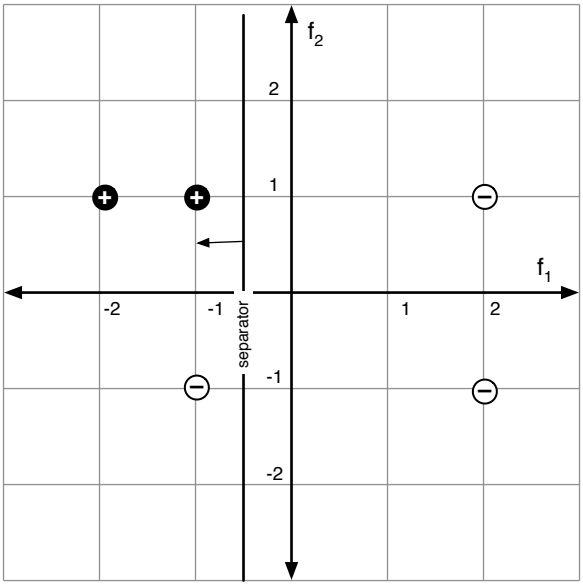
   By allowing the Lagrange multipliers to get too large

2. How can you reduce overfitting?

   Using cross-validation to choose the kernel parameters and the maximum value for the multipliers.

6.034 Quiz 4 Solutions, Spring 2006

1 Perceptron (20 points)



Data points are: Negative: (-1, -1) (2, 1) (2, -1) Positive: (-2, 1) (-1, 1)

Recall that the perceptron algorithm uses the extended form of the data points in which a 1 is added as the 0th component.

1. Assume that the initial value of the weight vector for the perceptron is $[0, 0, 1]$, that the data points are examined in the order given above and that the rate (step size) is 1.0. Give the weight vector after one iteration of the algorithm (one pass through all the data points):

   Only point $x_2 = (2, 1)$ *is misclassified. Using the extended form* $x_2' = (1, 2, 1)$*, we have*

   $$[0, 0, 1] * x_2' = +1$$

   We update the weight vector using the extended form (times $y_2 = -1$*):*

   $$w \leftarrow w + y_2 x_2'$$

   and we get the new weight vector

   $$w = [-1, -2, 0]$$

2. Draw the separator corresponding to the weights after this iteration on the graph at the top of the page.

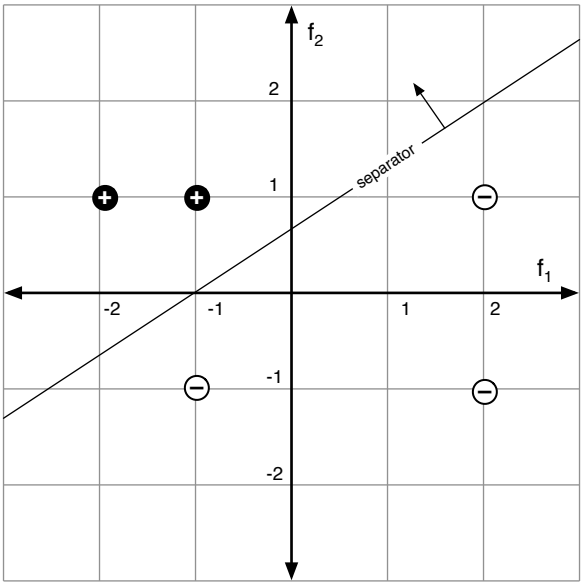
3. Would the algorithm stop after this iteration or keep going? Explain.

   No. The new weight vector misclassifies the negative point (-1, -1), whose margin will be +1.

4. If we add a positive point at (1,-1) to the other points and retrain the perceptron, what would the perceptron algorithm do? Explain.

   The data is no longer linearly separable and so the perceptron would loop forever.

3 Maximal Margin Linear Separator (20 points)



Data points are: Negative: (-1, -1) (2, 1) (2, -1) Positive: (-2, 1) (-1, 1)

1. Give the equation of a linear separator that has the maximal geometric margin for the data above. Hint: Look at this geometrically, don't try to derive it formally.

   (a) $w = [-2, 3]$

   (b) $b = -2$

2. Draw your separator on the graph above.

3. What is the value of the smallest geometric margin for any of the points?

   It's the margin for the support vectors divided by the magnitude of w, that is, $3/\sqrt{13}$

4. Which are the support vectors for this separator? Mark them on the graph above.

   The support vectors are (-1, 1), (-1, -1) and (2, 1)

4 SVM (20 points)

Assume that our training data is four 1-dimensional points, as follows:

index	x	y
1	-2	-1
2	-0.1	-1
3	0.1	1
4	1	1

1. Find the values of all the α_i that would be found by the SVM training algorithm, using a linear kernel. You should be able to do this without going through the Lagrangian minimization procedure. Think about the conditions for the optimization directly.

   Obviously, the support vectors are x_2 and x_3. We saw during the derivation of SVM's that $w(x_3 - x_2) = 2$, that is, $w(0.2) = 2$ and so the weight is $w = 10$. Since $w = 0.1\alpha_2 + 0.1\alpha_3$ and $\alpha_2 = \alpha_3$, we have that $w = 0.2\alpha_2$ and so

   $$\alpha_2 = \alpha_3 = 50$$

2. What would the offset be for these values of α_i?

   We know the margin of the support vectors must be 1, so $wy_3x_3 + b = 1$ so:

   $$b = 0$$

3. What if the value of C were set to 1? What would happen to the values of α_i and the offset? Explain.

   $C = 1$, means that the α_i cannot exceed 1. Since the α values for x_2 and x_3 are above 1, they would be capped at 1 and these points would no longer determine the location of the separator. Note that alpha$_1$ and α_4 would then become non-zero and determine the location of the separator. The new separator would be at $x = -0.5$ (midway between x_1 and x_4) and thus x_2 would be misclassified, but the geometric margin would increase.

5 Machine Learning (20 points)

For each of the statements below, indicate whether they are True or False and **briefly** justify your answer.

1. Given a data set and two alternative sets of weights for a neural network, we can pick between them using cross-validation.
   True or **False** Explain.

   False. Cross validation will construct multiple set of weights during its operation, so it can only be used to evaluate algorithms not particular hypotheses.

2. Given a data set and two values of C for an SVM, we can pick between them using cross-validation.
   True or **False** Explain.

   True. CV will compute how well, on average, the two values of C will work - over a variety of data sets derived from the original data set.

3. The Gaussian (RBF) kernel for SVMs is usually a better choice than the linear kernel.
   True or **False** Explain.

   False. The choice of kernel will depend on the data. If the data is close to being linearly separable, the simpler linear kernel will likely be a better choice. It is true that the Gaussian RBF is more powerful than the linear kernel and probably has more applications, so this received partial credit. Nevertheless, we want to stress that no particular method is inherently better for every applications. In machine learning, generality comes at a price.

4. We should train a neural net until its error on the training set is as low as possible.
   True or **False** Explain.

   False. This could lead to overfitting. We want to use a separate validation set to decide when to stop.

5. There is no value of K for which locally weighted averaging will produce exactly y^i when given x^i.
   True or **False** Explain.

   False. K=1 will produce this behavior.