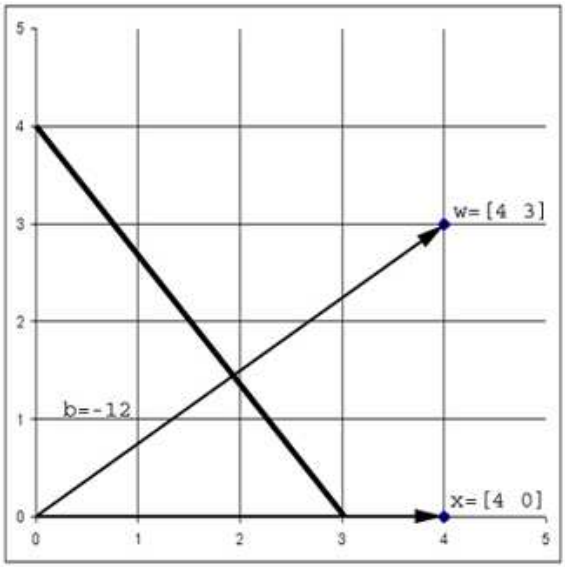
6.034 Artificial Intelligence
February 23, 2007 Recitation # 3

Machine Learning II ANSWERS

1 Linear Separators

1. In the graph below, we see a linear separator defined by a normal vector, **w = [4 3]** and an offset **b = -12**. Another way of representing this linear separator is to define a unit normal, $\hat{w}$, and a corresponding offset b' which represents the negated distance between the origin and the linear separator (recall that the distance between the origin and linear separator is always $-b/||w||$).



Calculate $\hat{w}$ and b' for this linear separator:

$\hat{w} = [\frac{4}{5}, \frac{3}{5}]$

$b' = \frac{-12}{5}$

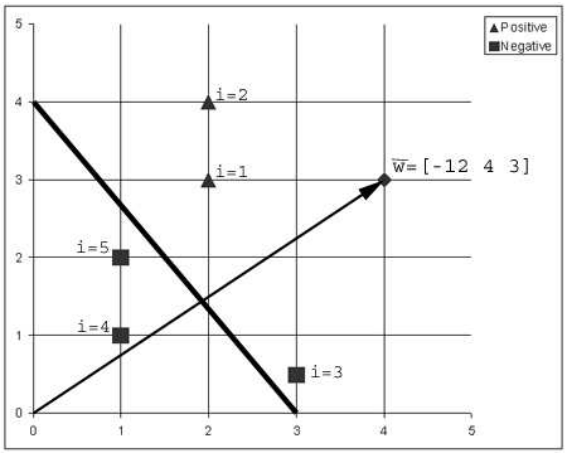
2. In the graph above, point **x = [4 0]** should be classified as a **+1**. To calculate its margin, we project it onto the normal, w, add the offset, b, and multiply it by the desired output, in this case, +1. For this calculation, we will use an alternative notation for the vectors: $\overline{w}$ which, in this case, is the vector [-12 4 3] and $\overline{x}$ which is the vector [1 4 0].

margin = $(+1) \cdot (12 \cdot 1 + 4 \cdot 4 + 3 \cdot 0) = 4$

Is this point classified correctly given our linear separator? **Yes.**

2 Perceptron Algorithm

1. In the graph below, we see the same linear separator from section 1 but now we also have 5 data points which should be classified as either positive (+1) or negative (-1).

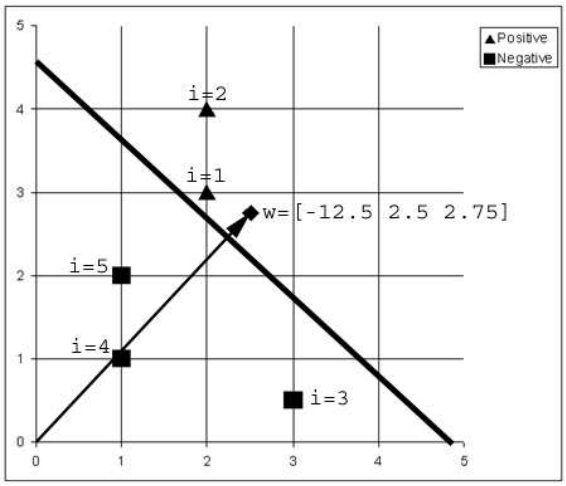


Simulate one iteration of the perceptron algorithm with a learning rate of **0.5**:

$\overline{w} =$ [-12 4 3]
$\overline{w}' = \overline{w} + \eta * y^i * x^i$

i	x_0^i	x_1^i	x_2^i	y^i	$\overline{w}$ (new values)
1	1	2	3	+1	$[-12, 4, 3]$
2	1	2	4	+1	$[-12, 4, 3]$
3	1	3	0.5	-1	$[-12.5, 2.5, 2.75]$
4	1	1	1	-1	$[-12.5, 2.5, 2.75]$
5	1	1	2	-1	$[-12.5, 2.5, 2.75]$

2. The new linear separator defined by the new weight vector correctly classifies all the points (see graph below).



- How does the learning rate affect the new separator we get after each iteration of the perceptron algorithm? What would happen if we used a learning rate of 0.001 versus 10?

  When eta is 0.001, the algorithm takes 148 iterations to converge, yielding the linear separator $[-12.1, 3.6, 2.9]$**.**

  When eta is 10, it takes 10 iterations and yields $[-72, -16, 43]$**.**

- In the table below are the alpha values if we ran the perceptron algorithm with an initial weight vector of all 0's. Using the alpha values write the resulting output from the dual-form perceptron algorithm.

i	# time incorect (α)	x	y
1	9	[1 2 3]	+1
1	0	[1 2 4]	+1
1	2	[1 3 0.5]	-1
1	6	[1 1 1]	-1
1	8	[1 1 2]	-1

The dual form is $\sum_i \alpha_i y_i x_i = (9 \cdot [1,2,3] + 0 \cdot [1,2,4] - 2 \cdot [1,3,0.5] - 6 \cdot [1,1,1] - 8 \cdot [1,1,2] = [-7,-2,4]$**.**

3 Nearest Neighbor

The Matsakis Macintosh School has found that many teachers just aren't happy and tend to leave. Trying to reduce turnover, the administration decides to poll the current teaching staff in an effort to predict if candidate teachers will be happy or not before offering them a position. After studying the data, the superintendent decides that two factors are most important: the difference (D) between an employee's previous salary and current salary (a number which is always positive, since everyone earns less at MatMac than he or she did before, and ranges between 0-20K), and the number of years (Y) he or she has worked as a teacher (ranging between 0-20).

Figure 1 shows the data in graphical form.

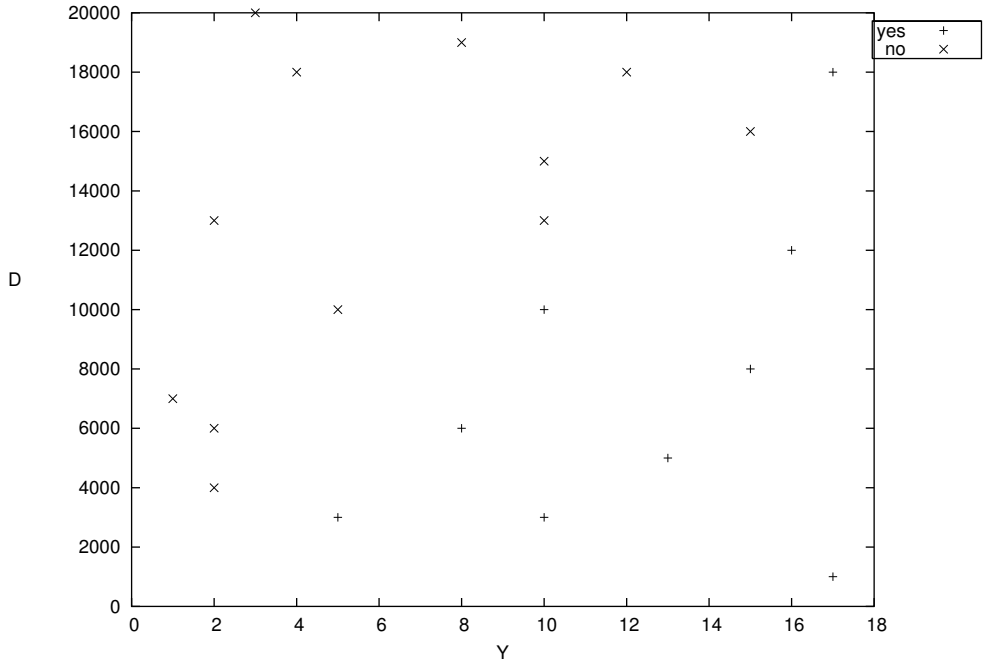


Figure 1: Teacher data, no scaling.

1. The Matsakis School interviews a candidate who currently earns $33,000 and will be offered $30,000 to teach there. He has 1 year of prior teaching experience.

Plot this new point on the graph. Without considering any particular distance metric, what is your intuition for how this new candidate should be classified? Why?

One possible answer: the new point should be classified as "no" because everyone in the data set who has taught for fewer than 4 years is unhappy.

Now, let's try using the **naive Euclidean distance** to classify the new point:

$$D(x^j, x^k) = \sqrt{\sum_j (x_j^i - x_j^k)^2}$$

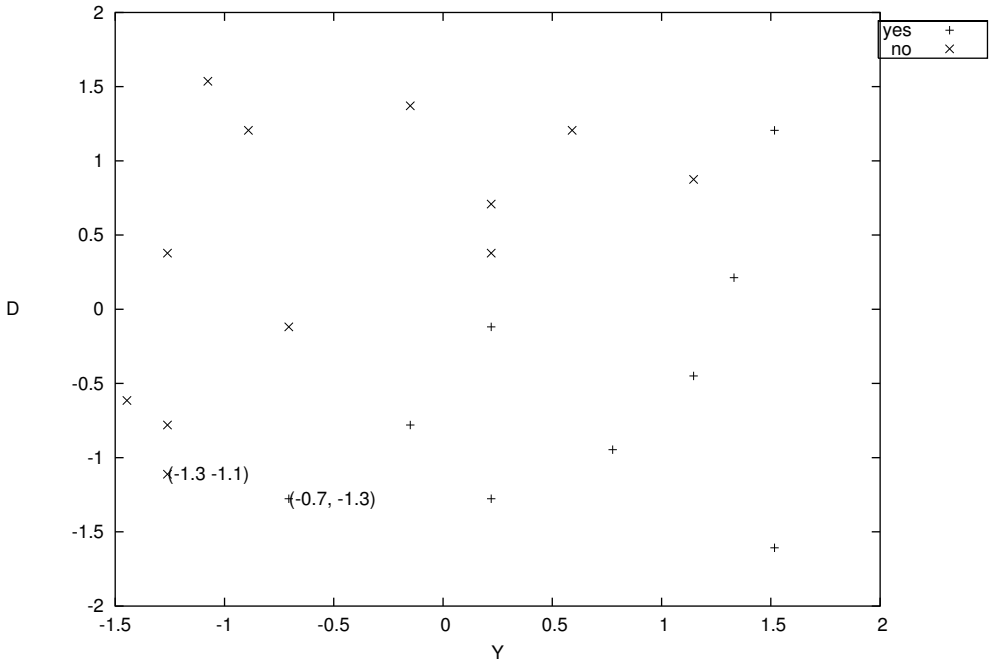


Figure 2: Teacher data, normalized.

If we use this distance metric with the 1-nearest neighbor algorithm, how is the new point classified? Why?

If we calculate the distance between the new point (1,3000) the nearest "yes" (5,3000) and the nearest "no" (2,4000) points, we find the distances are 4 and approx. 1000, respectively. So we predict that the new candidate will be happy.

2. We can rescale (or **normalize**) the features by computing the **average value** $\bar{x}$ and **standard deviation** σ:

$$x' = \frac{x - \bar{x}}{\sigma_x}$$

If we calculate the average value and the standard deviation for each of our features in the data set above, we get the following values:

$$\begin{aligned} \bar{y} &\approx 8.8 \\ \sigma_y &\approx 5.4 \\ \bar{d} &\approx 10,714 \\ \sigma_d &\approx 6043 \end{aligned}$$

- Plotting the resulting normalized data, we get the graph in Figure 2. What is the value of the point we are interested from above – (1, 3000) – after scaling, rounded to the nearest tenth?
- Plot this point on the graph. Now, use the naive Euclidean distance metric to determine the distance from the new point to the nearest positive and nearest negative points on the graph. The values of these points are given to you in the plot, rounded to the nearest tenth.

- Using 1-nearest neighbor, what is the class of the point after scaling? Why?

After scaling, the point we are interested in from above – (1, 3000) – is (-1.4, -1.3), rounded to the nearest tenth.

The class of the point is now "no" because the new point is closer to the "no" point (.22) than the "yes" point (.7) in Euclidean space, after scaling.

3. Another form of **scaling** can emphasize the relative importance of the features in the distance metric. For instance, we may want to emphasize the importance of the Y feature in our data. Recall that prior to normalizing our data, the vastly different ranges of the Y and D features caused problems for our naive Euclidean distance metric. Below, we show four graphs. Figure 3 shows the data with no scaling, but with axes drawn at the same scale. Here, we can see quite clearly that the Y feature has virtually no impact on classification. Figures 4, 5, and 6 show the data with the Y feature scaled by 200, 500, and 1000, respectively.
Of these three graphs, list all those that are *sufficient* to classify our new data point from above – (1, 3000) – as a "no"? The scaled coordinates of the closest "yes" and "no" points are labeled on each graph.

The distance from (200, 3000) to (1000,3000) – the closest "yes" – is 800, while the distance to (400,4000) – the closest "no" – is 1166. So scaling by 200 is not sufficient. The other two scaling factors are, as can be seen quite readily in the graphs.

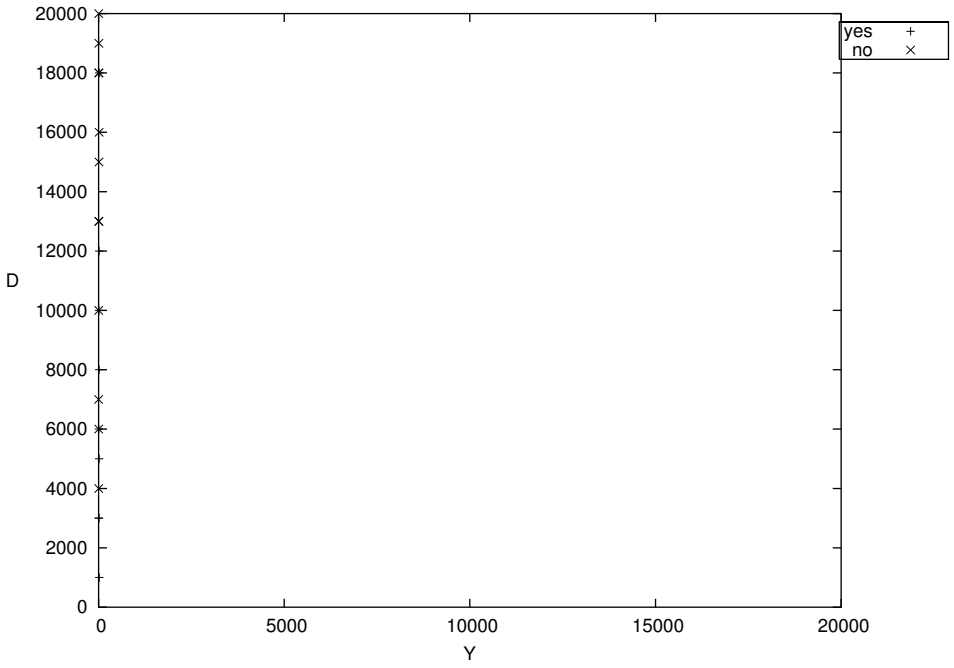


Figure 3: Original teacher data with axes drawn at the same scale.

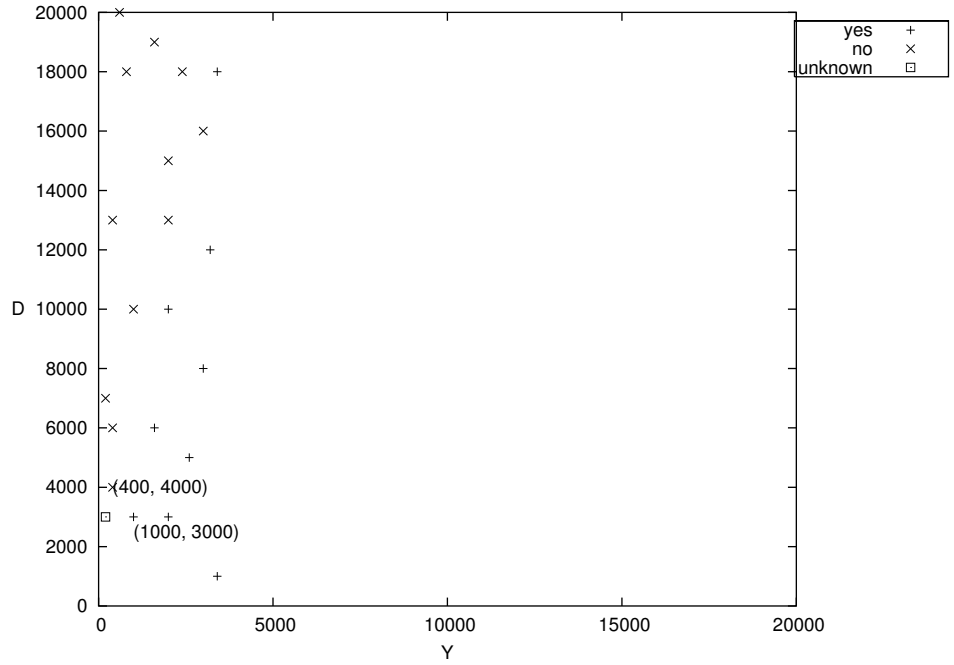


Figure 4: Teacher data with feature Y scaled by 200.

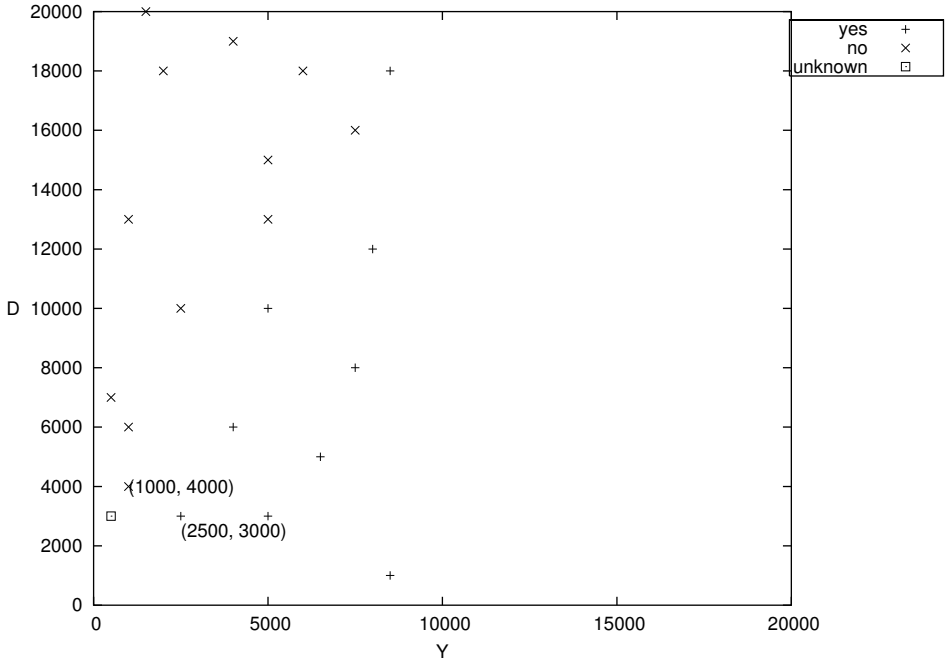


Figure 5: Teacher data with feature Y scaled by 500.

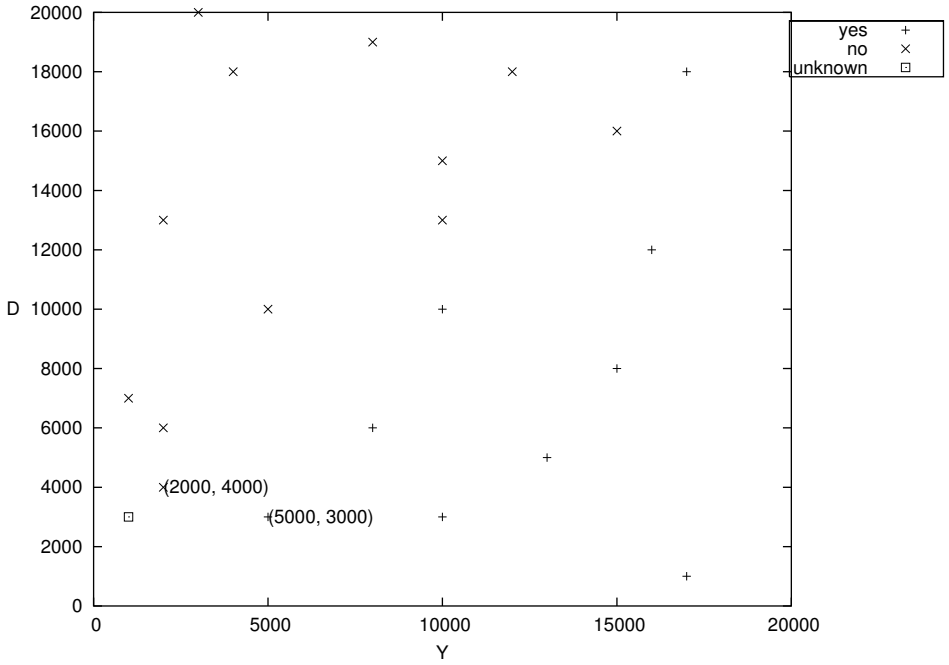


Figure 6: Teacher data with feature Y scaled by 1000.

4. We can perform **leave-one-out cross-validation** to determine whether the scaling factors we just said were sufficient to classify our unknown point as a "no" are good given our training data. To do this, we remove each point, one by one, from the data, run our nearest neighbor algorithm, and see if we get the correct class. Figure 7 shows the original data with the Y feature scaled by 1000. Is this scaling factor good enough to correctly classify the training data? Why or why not?

No. Many of the points that are classified as "yes" would be classified as "no" when running leave-one-out cross-validation. Also, one of the "no" points would be misclassified as a "yes."

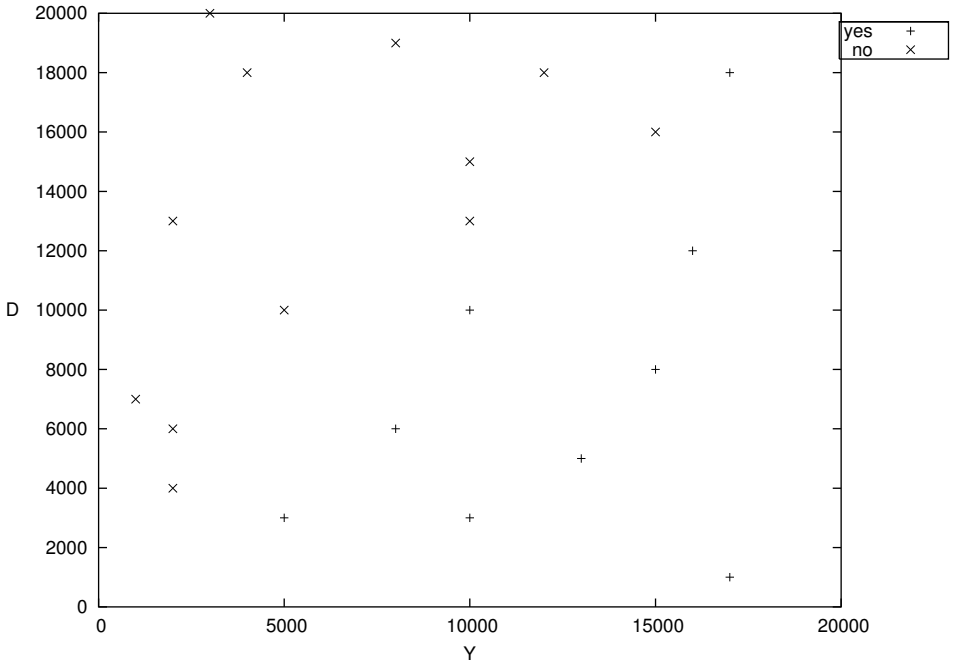


Figure 7: Teacher data with feature Y scaled by 1000.

4 Return of the Decision Tree

1. We can build a decision tree for our teacher data by considering the average entropy of possible splits for the two features. Figure 8 shows the splits we will consider at the top level of our decision tree. Just looking at the splits, is there one that you think may be the best for our data at the top level of the tree? Why?

Looking at the data, we see that if we choose the split Y < 4.5, we will have a leaf node on the right branch (i.e., the "yes" branch). This may be a good split.

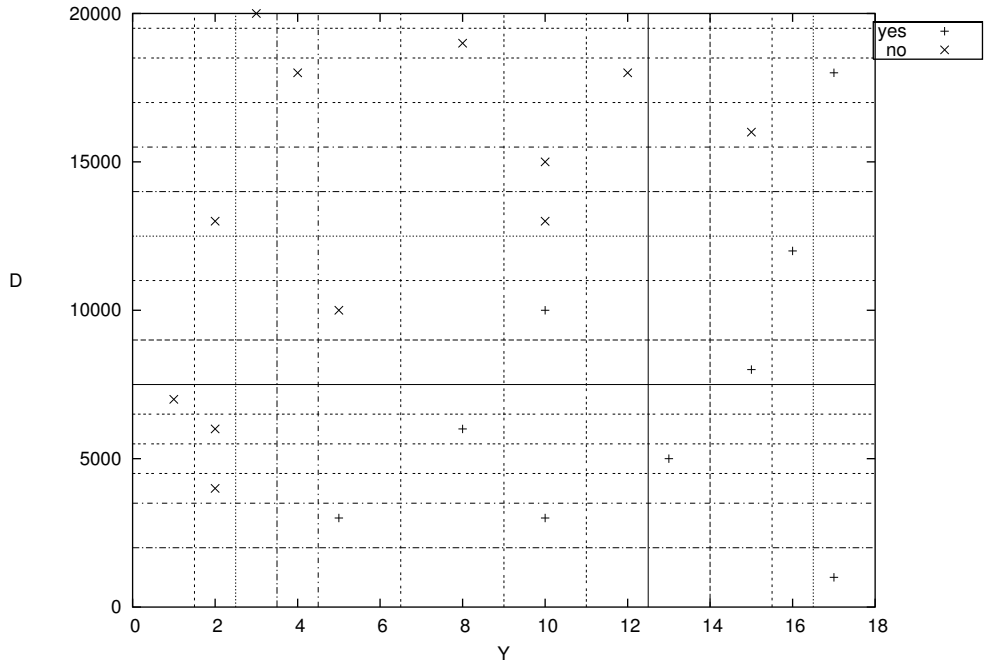


Figure 8: Using decision trees for teacher data.

Table 1 shows the average entropy in the Y dimension for each of these splits, while Table 2 shows the average entropy in the D dimension. One row in each table has been left blank; be sure to fill those in. Which split should we choose for the top level of the tree?

The split with the lowest average entropy is Y < 4.5, as predicted.

2. On the graph in Figure 8, draw the line that represents the split we just chose. How many splits in the Y dimension would we consider in the next iteration of the algorithm? How many in the D dimension?

7 in the Y dimension and 14 in the D dimension.

3. Do we ever need to worry about scaling when we are using decision trees?

No, we don't. When we were using the Euclidean distance metric to determine nearness, the units mattered, but here, we are taking neither distance nor units into consideration.

Y < x	No-True	Yes-True	No-False	Yes-False	AE
1.5	1	0	11	9	.95
2.5	4	0	8	9	.81
3.5	5	0	7	9	.75
4.5	6	0	6	9	.69
6.5	7	1	5	8	.80
9	8	2	4	7	.84
11	10	4	2	5	.86
12.5	11	4	1	5	.78
14	11	5	1	4	.85
15.5	12	6	0	3	.79
16.5	12	7	0	2	.86

Table 1: Average entropy for top-level splits in Y dimension.

D < y	No-True	Yes-True	No-False	Yes-False	AE
2000	0	1	12	8	.92
3500	0	3	12	6	.79
4500	1	3	11	6	.91
5500	1	4	11	5	.85
6500	2	5	10	4	.86
7500	3	5	9	4	.91
9000	3	6	9	3	.86
11000	4	7	8	2	.84
12500	4	8	8	1	.74
14000	5	9	6	1	.82
15500	6	9	5	1	.88
17000	7	9	4	1	.93
18500	9	10	2	0	.96
19500	10	10	1	0	.95

Table 2: Average entropy for top-level splits in D dimension.

5 Regression

1. Which of the following are regression problems, and which are classification problems?

Problem	R/C
Predicting the gas mileage of an automobile based on its weight in pounds and number of cylinders.	**Regression**
Labeling the part of speech of a word based on its position in a sentence and the part of speech of the preceding word.	**Classification**
Determining the amount of force a robot needs to apply to move an object based on the weight of the object and the direction of movement.	**Regression**
Mapping a segment of audio data to a phoneme based on the value of the fundamental frequency.	**Classification**

2. How could we turn our teacher classification problem into a regression problem?

One possible answer: We could poll the teachers and ask them to rank their satisfaction on a scale from 1-5. Then, when we do prediction, we try to figure out where on the scale the new candidate will fall.

3. Can we use the nearest neighbor algorithm to solve a regression problem? What about decision trees?

Yes. For nearest neighbor, we can take the k nearest neighbors and average the output associated with them, or we can do locally-weighted averaging using a kernel. If we use decision trees, we call them regression trees and use variance instead of average entropy to determine where to make our splits.