CS273a Midterm Exam

Machine Learning & Data Mining: Fall 2013

Thursday November 7th, 2013

Your name: SOLUTIONS

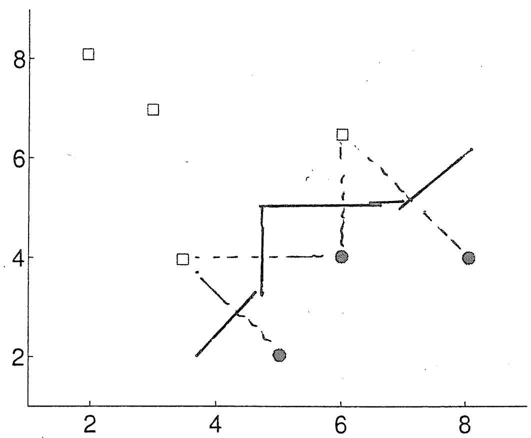
Name of the person in front of you (if any):

Name of the person to your right (if any):

- Total time is 1:15. READ THE EXAM FIRST and organize your time; don't spend too long on any one problem.
- Please **write clearly** and **show all your work**.
- If you need clarification on a problem, please raise your hand and wait for the instructor to come over.
- Turn in any scratch paper with your exam.

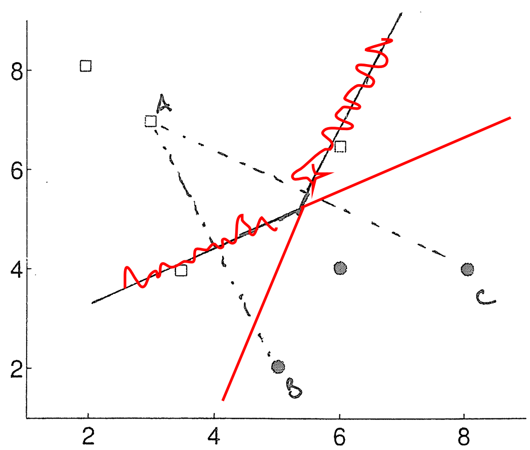
Problem 1: (9 points) K-Nearest Neighbor Classification

Consider the following set of training data, consisting of two-dimensional real-valued features and a binary class value, for a k-nearest-neighbors classifier. Positive data are shown as circles, negative as squares.



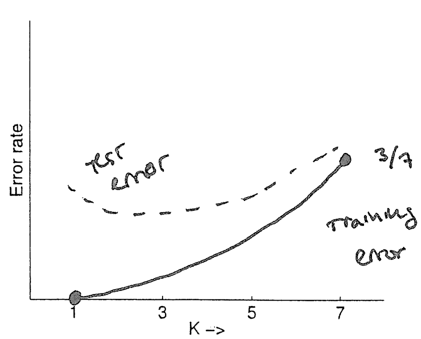
(1) Sketch the decision boundary for $k = 1$. Show your work and justify your answer in a few sentences (2-3).

Decision boundary is piecewise linear, perpendicular & halfway along lines connecting the data points (dashed)



(2) Sketch the decision boundary for $k = 5$, in the relevant part of the feature space (i.e., near the training data). Again, show your work and justify your answer in a few sentences.

In the lower right, we decide +1 until the 3rd (-1) point A is closer than one of the two positives (B, C).



(3) Sketch the basic shape you would expect to see for the error rate on training data, and on test data, as a function of increasing $k = 1, \ldots, 7$. For the training error rate, **indicate the values** (error rates) of the endpoints ($k =$ 1 and $k = 7$).

At $k=0$, training error rate $= 0$

$k=7$, " $= 3/7$

Problem 2: (10 points) Under- and Over-fitting

Circle one answer for each:

When training a linear classifier with gradient descent, we decrease the maximum number of iterations performed by the algorithm. This will make it **more** **equally** **less** likely to overfit the data.

Suppose we are using a *weighted* neighbor regression model, where for a test point x we assign the data weights $w^{(i)} = \exp(-c\|x - x^{(i)}\|)$. Increasing c will make it **more** **equally** **less** likely to overfit the data.

When training a decision tree, we had a parameter (`minParent`) that forced us to *never* split a node if there were fewer than `minParent` data in that node. Suppose we decrease `minParent` from its current value. This will make it **more** **equally** **less** likely to overfit the data.

Adding features to a decision tree classifier will make it **more** **equally** **less** likely to overfit the data.

For the next several questions, suppose that we are using a linear classifier, and we currently believe our model to be **overfitting**. We decide to increase the number of training data used by our learner. Choose one answer for each part:

Training error will most likely **increase** **stay the same** **decrease.**

Test error will most likely **increase** **stay the same** **decrease.**

The VC dimension of our learner will most likely **increase** **stay the same** **decrease.**

Now suppose we believe our model is **underfitting**. We again increase the number of training data. Choose one answer for each part:

Training error will most likely **increase** **stay the same** **decrease.**

Test error will most likely **increase** **stay the same** **decrease.**

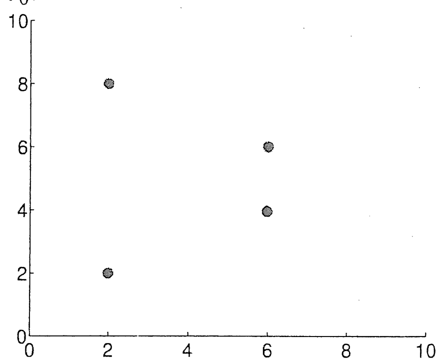
The VC dimension of our learner will most likely **increase** **stay the same** **decrease.**

(Hint: think about the relationship between training and test error rates in each regime, what this will mean for our performance on the newly added data, and how much these new data will influence our model parameters.)

Problem 3: (8 points) Cross-validation and Linear Regression

Consider the following data points, copied in each part. We wish to perform linear regression to minimize the mean squared error of our predictions.

(a) Compute the leave-one-out cross-validation error of a zero-order (constant) predictor, $\hat{y}(x) = \theta_0$:



Best-MSE constant predictor = mean of the training points.

⇒ predict 4

⇒ SE: $(4)^2$

⇒ SE $= (4)^2$

⇒ predict $14/3$

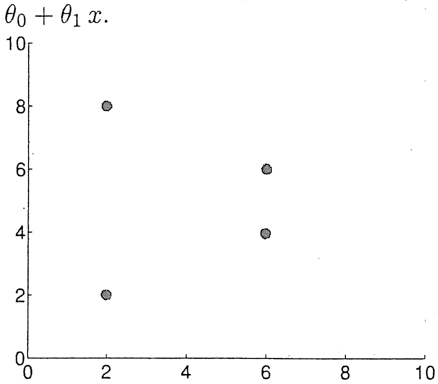
⇒ SE $= (4/3)^2$

⇒ SE $= (4/3)^2$

$$\Rightarrow MSE = \frac{1}{4}\left(4^2 + 4^2 + (4/3)^2 + (4/3)^2\right)$$

$$= 8 + 8/9$$

(b) Compute the leave-one-out cross-validation error of a first-order (linear) predictor, $\hat{y}(x) = \theta_0 + \theta_1 x$.



Best MSE linear predictor will either pass through the data point (if only value for that x) or midpoint (if 2 data for that x)

⇒ SE: $(6)^2$

⇒ SE $= 6^2$

⇒ SE $= 2^2$

SE $= 2^2$

$$MSE = \frac{1}{4}\left(6^2 + 6^2 + 2^2 + 2^2\right)$$

$$= 20$$

Problem 4: (11 points) Gradient Descent

Suppose that we wish to train the following non-linear regression model:

$$\hat{y}(x) = \exp(\, w_0 + w_1 x_1 + w_2 x_2 \,)$$

(a) Given data $D = \{(x^{(i)}, y^{(i)})\}$, write down the mean-squared error loss function $J(w)$ for the model, and compute the gradient of $J(w)$ with respect to the weights.

$$J(w) = \frac{1}{m}\sum_{i=1}^{m} (y^i - \hat{y}(w, x))^2 = \frac{1}{m}\sum (y^i - \exp(w_0 + w_1 x_1^i + w_2 x_2^i))^2$$

$$\frac{\partial J}{\partial w_k} = \frac{1}{m}\sum (-1)(y^i - \exp(\cdots)) \cdot \exp(\cdots) \cdot x_k^i$$

$$\Rightarrow \nabla_w J = \frac{1}{m}\sum \underbrace{(-1)(y^i - \hat{y}(w, x^i)) \cdot \hat{y}(w, x^i)}_{\text{Scalar}} \cdot \underbrace{[1 \;\; x_1^i \;\; x_2^i]}_{\text{vector.}}$$

(b) Suppose now that, instead of feature 2, we have a transformed version of feature 1, i.e.,

$$\hat{y}(x) = \exp(\, w_0 + w_1 x_1 + w_2 h_2(x_1)\,)$$

where $h_2(x_1) = \log(\, x_1 + \alpha\,)$, and we wish to also update α using gradient descent. Give the derivative of J with respect to α.

By chain rule:

$$\frac{\partial J}{\partial \alpha} = \frac{1}{m}\sum (y^i - \hat{y}(w, x^i)) \cdot \hat{y}(w, x^i) \cdot \frac{\partial}{\partial \alpha}[w_0 + w_1 x_1^i + w_2 x_2^i]$$

$$= \frac{1}{m}\sum (y^i - \hat{y}) \cdot \hat{y} \cdot w_2 \frac{1}{x_1 + \alpha}$$

(c) Suppose that, instead of using an exponentiated linear function as described above, we transformed our target y by taking its log, $\tilde{y} = \log(y)$, and performed a standard linear regression with target $\tilde{y}$. What loss function is this minimizing? How would its predictions likely differ from the nonlinear regression above? (Hint: comment on the relative magnitude of errors for different values of y.)

method A

method B.

They differ in the relative importance they place on errors at ~~high~~ large y vs small y.

(Also, Method B can be trained in closed form, which is nice.)

Suppose $y=1$ and $\hat{y}=2$ - for method A, this is the same error as $y=20$ and $\hat{y}=21$.

For method B, these errors are $(y=1 \Rightarrow \tilde{y}=0,\ \hat{y}=\log(2))$ vs $(\tilde{y}=\log(20),\ \hat{y}=\log(21))$

Much larger squared error than

Problem 5: (12 points) Decision Trees

We plan to use a decision tree to predict an outcome y using four features, $x_1, \ldots, x_3$. We observe nine training patterns, each of which we represent as $[x_1, x_2, x_3]$ (so, "010" means $x_1 = 0$, $x_2 = 1$, $x_3 = 0$). We observe the training data,

$y = 0:$ [001], [010], [010], [110]
$y = 1:$ [000], [011], [101], [111]

(Note that one feature vector is observed twice.)

You may find the following values useful (although you may also leave logs unexpanded):

$\log_2(1) = 0$ $\log_2(2) = 1$ $\log_2(3) = 1.59$ $\log_2(4) = 2$ $\log_2(5) = 2.32$
$\log_2(6) = 2.59$ $\log_2(7) = 2.81$ $\log_2(8) = 3$ $\log_2(9) = 3.17$ $\log_2(10) = 3.32$

In case of ties, we prefer to use the feature with the smaller index (x_1 over x_2, etc.) and prefer to predict class 1 over class 0.

(a) What is the entropy of y?

$p(y=1) = \frac{1}{2} \Rightarrow H(y) = 1$ bit

$x = a$: $y=0$ examples | $y=1$ examples

(b) Which variable would you split first? Justify your answer.

if $x_1=0$

$y=0$	$y=1$
001	000
010	011
010	

$x_1=1$

$y=0$	$y=1$
110	101
	111

$x_2=0$

$y=0$	$y=1$
001	000
	101

$x_2=1$

$y=0$	$y=1$
010	011
010	111
110	

$x_3=0$

$y=0$	$y=1$
010	000
010	
110	

$x_3=1$

$y=0$	$y=1$
001	011
	101
	111

By inspection, x_3 is the best (it has lower entropy in both cases than the other splits)

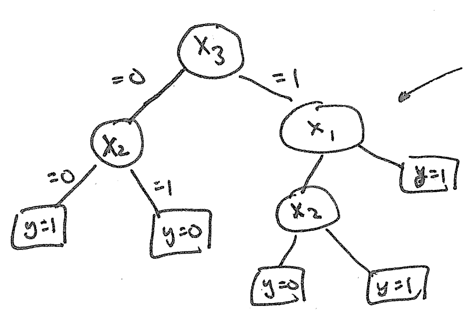
(c) What is the information gain of the variable you selected in part (b)?

$$IG = H(y) - \left[p(x_3=0)\, H(y|x_3=0) + p(x_3=1)\, H(y|x_3=1) \right]$$

$$= 1 - \tfrac{1}{2} H(3/4) - \tfrac{1}{2} H(3/4)$$

$$= 1 + \tfrac{3}{4} \log \tfrac{3}{4} + \tfrac{1}{4} \log \tfrac{1}{4}$$

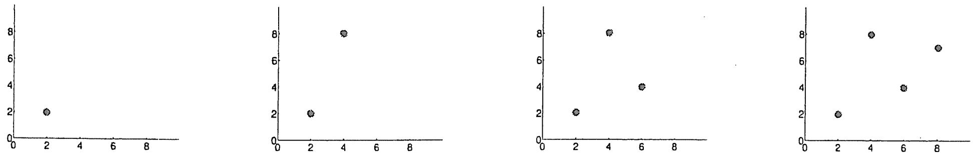
(d) Draw the rest of the decision tree learned on these data.



could pick either, but instructions say to prefer x_1

Problem 6: (8 points) Shattering & VC Dimension

Which of the following examples can be shattered by each of the learners below? (You do not have to formally prove, but justify your answer briefly.)



(a) (b) (c) (d)

For the two learners, $T[z]$ is the sign threshold function, $T[z] = +1$ for $z \geq 0$ and $T[z] = -1$ for $z < 0$. The learner parameters a, b, c are real-valued scalars, and each data point has two real-valued input features x_1, x_2.

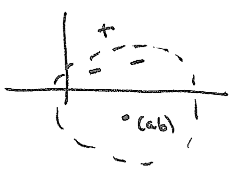
(a) $\hat{y} = T[\,(x_1 - a)^2 + (x_2 - b)^2 + c\,]$ (note: uses both x_1, x_2) – predicts +1 outside a circle at (a,b) with radius $\sqrt{-c}$

Ans: (a), (b), and (c) – but not (d).

(a) – easy

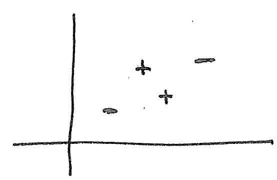
(b) – move center of the circle near "–" datum

(c)



but

(d)



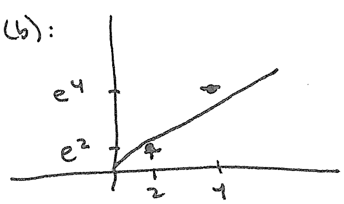
No circle can include the two negative examples & not include the positive.

(b) $\hat{y} = T[\,a x_1 + b \exp(x_1)\,]$ (note: uses only x_1!) – A 2-feature perceptron w/ no intercept (bdry passes through the origin).

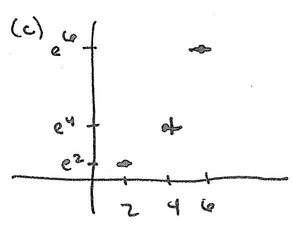
Feature 2 is $\exp(x_1)$, but that's ok; we just need to see where the data points move to.

Ans: (a), (b) but not (c), (d)

Ex (b):



Ex (c)



no line that passes through the origin and can separate these examples.