CS273a Midterm Exam

Machine Learning & Data Mining: Fall 2012

Thursday November 1st, 2012

Your name: SOLUTIONS

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

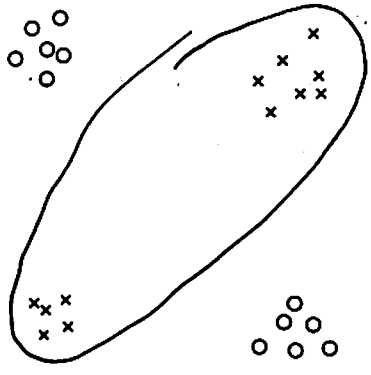
Name of the person to your right (if any):

- 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.

(This page intentionally left blank)

Problem 1: (12 points) Separability

For each of the following examples of training data, sketch a classification boundary that separates the data. State whether or not the data are linearly separable, and if not, give a set of features that would allow the data to be separated.

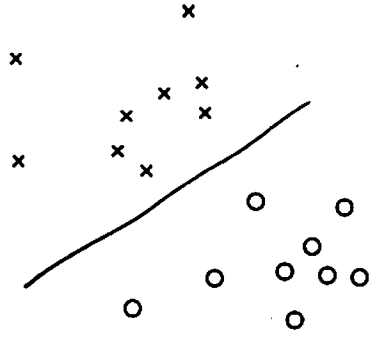


NOT linearly separable

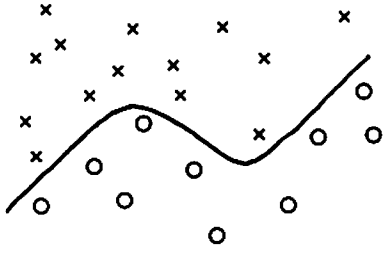
could use eg.

$[x_1 \; x_2 \; x_1x_2 \; x_1^2 \; x_2^2]$

to produce the ellipsoidal decision boundary



yes, linearly separable



No, not linearly separable

from the decision boundary, it looks like

$x_2 = a + bx_1 + cx_1^2 + dx_2^3$

so features

$[1 \; x_1 \; x_2 \; x_1^2 \; x_1^3]$ are probably sufficient.

Problem 2: (13 points) Under- and Over-fitting

(a) Suppose that we train a classifier, and discover that it achieves zero (training) error. Are we likely to be over-fitting, under-fitting, neither, or do we need more information? Explain (1-2 sentences).

Need more information – it is impossible to check overfitting from only the training data error; we need validation data or cross-validation.

It could be that we are overfit & have memorized the data; or, it could be that the data are easy to predict & our classifier is very good.

(b) **Circle one answer for each:**

Adding features to a linear classifier will make it the data.	(more)	equally	less likely to overfit
Increasing the regularization parameter for a linear classifier will make it (less) likely to overfit the data.	more	equally	
Increasing the step size in gradient descent for a linear classifier will make it less likely to overfit the data. (May depend on stopping criteria...)	more	(equally)	
Increasing the value of k in a k-nearest neighbor classifier will make it (less) likely to overfit the data.	more	equally	
Increasing the number of hidden nodes in a neural network will make it less likely to overfit the data.	(more)	equally	

Problem 3: (10 points) Regression

Suppose that we have training data $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$ and we wish to predict y using the model:

$$\hat{y}(x) = a \log(x + b) + c$$

(a) Is this a linear or nonlinear regression model. Why?

Nonlinear - it is a non-linear function of parameter b.

(b) Write the mean squared error cost function for our predictor.

$$J(a,b,c) = \frac{1}{N} \sum_i (y^i - \hat{y}^i)^2$$

$$= \frac{1}{N} \sum_i \left(y^i - (a \log(x^i + b) + c)\right)^2$$

(c) Compute its gradient with respect to the parameters a, b, and c.

$$\frac{\partial}{\partial a} J = \frac{2}{N} \sum_i (y^i - \hat{y}^i)(-\log(x^i + b)) \quad \text{where} \quad \hat{y}^i = (a \log(x^i + b) + c);$$

$$\frac{\partial}{\partial b} J = \frac{2}{N} \sum_i (y^i - \hat{y}^i)\left(-a \frac{1}{x^i + b}\right)$$

$$\frac{\partial}{\partial c} J = \frac{2}{N} \sum_i (y^i - \hat{y}^i)(-1)$$

$$\nabla J = \left[\frac{\partial}{\partial a} J \quad \frac{\partial}{\partial b} J \quad \frac{\partial}{\partial c} J\right]$$

$$= \frac{-2}{N} \sum_i (y^i - \hat{y}^i) \left[\log(x^i + b) \quad \frac{a}{x^i + b} \quad 1\right]$$

scalar · vector

Problem 4: (6 points) Optimization

(a) Give pseudocode for a stochastic (online or incremental) gradient descent algorithm to optimize the model in Problem 2. You do not need to have solved for the gradient $\nabla J(a, b, c)$ to do this; just assume it can be computed. Explain all the parameters used by your algorithm.

Initialize parameters θ to something; $J = \infty$

while (! done) {

$J^{old} = J$.

for $i = 1 \ldots N$

$J_i = (y^i - \hat{y}^i)^2$

∇J_i is the gradient of J_i

$\theta \leftarrow \theta - \alpha \nabla J_i$ where α is a step size.

end

compute $J = \frac{1}{N} \sum (y^i - \hat{y}^i)^2$

done = true if:

① too many iterations already or

② J hasn't changed from the last iteration. $(|J - J^{old}| < \epsilon)$

}

(b) Explain the difference between batch and stochastic gradient descent (1-3 sentences). Name one advantage for each.

Define $J_i(a,b,c)$ to be the loss on data point i, eg $(y - \hat{y}^i)^2$

$\Rightarrow J(a,b,c) = \frac{1}{N} \sum_i J_i(a,b,c)$

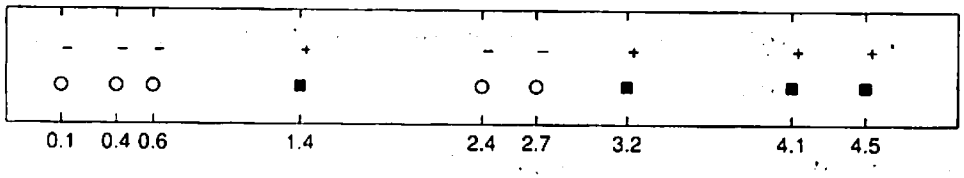
Batch gradient descent uses ∇J, the gradient of the loss on all data, for each update;

SGD uses ∇J_i, the gradient on each point i, to update sequentially.

Batch GD will decrease the overall loss J, so its easier to check convergence and "smoother".

SGD is noisier (more random), but in practice tends to optimize the parameters much faster, especially when N is very large.

Problem 5: (12 points) Cross-validation and Nearest Neighbor



Using the above data with one feature x (whose values are given below each data point) and a class variable $y \in \{-1, +1\}$, with squares indicating $y = +1$ and circles $y = -1$ (the sign is also shown above each data point for redundancy), answer the following:

(a) Compute the leave-one-out cross-validation error of a 1-Nearest-Neighbor classifier. In the case of any ties, select the left-most neighbor at the same distance as the nearest.

✓ ✓ ✓ × ✓ ✓ × ✓ ✓

⇒ 2/9 error rate.

(b) Compute the leave-one-out cross-validation error for a 3-Nearest-Neighbor classifier.

✓ ✓ ✓ × × × × ✓ ✓

⇒ 4/9 error rate

(c) Compute the leave-one-out cross-validation error for a 8-NN classifier. In the case of a tie, predict class +1.

tie tie

× × × × × × × × ×

⇒ 9/9 error rate (!)

Problem 6: (14 points) VC Dimension

(a) Describe VC dimension in your own words, in a few (2-4) sentences.

A classifier can shatter a set of points if it can learn to produce any pattern of class values on those points.

The VC dimension is the largest # of points that can be arranged such that they can be shattered.

(b) Give an example of a model in which the VC dimension is not equal to the number of parameters.

From HW (or similar) $T[a(bx_1 + cx_2 + d)]$

- has four parameters, but "a" doesn't really help in any way.

(c) Circle one answer for each:

Increasing the amount of training data in a linear classifier will **decrease** the VC dimension. — **increase** / (**not change**)

Increasing the number of features used in a linear classifier will **decrease** the VC dimension. — (**increase**) / **not change**

Increasing the regularization parameter for a linear classifier will **decrease** the VC dimension. — **increase** / (**not change**)

Exponentiating feature 1 before training (e.g., `x(:,1) = exp(x(:,1));`) a linear classifier will **increase** (**not change**) **decrease** the VC dimension.

Problem 7: (12 points) Support Vector Machines

Consider a linear classifier, $T(wx^T + b)$, where $x = [x_1, \ldots, x_d]$ is a d-dimensional feature vector, $w = [w_1, \ldots, w_d]$ are the coefficients, and b is the constant coefficient.

In class, I described how we could optimize a SVM written in constraint form,

$$\min_w \|w\|^2 \qquad s.t. y^{(i)}(wx^{(i)} + b) \geq 1$$

by optimizing w along with a set of Lagrange multipliers α:

$$J(w, \alpha) = \min_w \max_{\alpha \geq 0} \|w\|^2 + \sum_i \alpha_i (1 - y^{(i)}(wx^{(i)^T} + b))$$

(a) By solving $\nabla_w J(w, \alpha) = 0$, show that the optimal value of w is

$$w^* = \sum_i \alpha_i y^{(i)} x^{(i)}$$

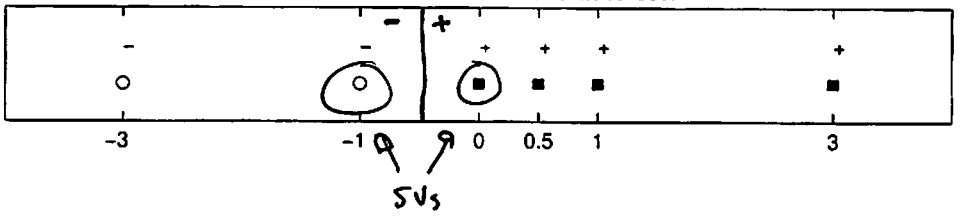
(Hint: just take the derivative and solve for w_1 and argue symmetry.)

$$\frac{\partial}{\partial w_1} J(w, \alpha) = w_1 + \sum_i -\alpha_i y^{(i)} x_1^{(i)} = 0$$

$$\Rightarrow \quad w_1 = \sum_i \alpha_i y^{(i)} x_1^{(i)} \qquad w = [w_1 \; w_2 \; w_3 \; \ldots]$$

$$\Rightarrow \quad \underline{w} = \sum_i \alpha_i y^{(i)} \underline{x}^{(i)}$$

(b) For the following data, sketch the decision boundary, identify the support vectors, and give the value of w and b for a linear SVM trained on the data set.



$w x + b = 1 \;$ @ $x = 0$

$w x + b = -1 \;$ @ $x = -1$

$\Rightarrow \quad b = 1$

$w = +2$

CS273a Midterm Exam

Machine Learning & Data Mining: Fall 2013

Thursday November 7th, 2013

Your name: SOLUTIONS

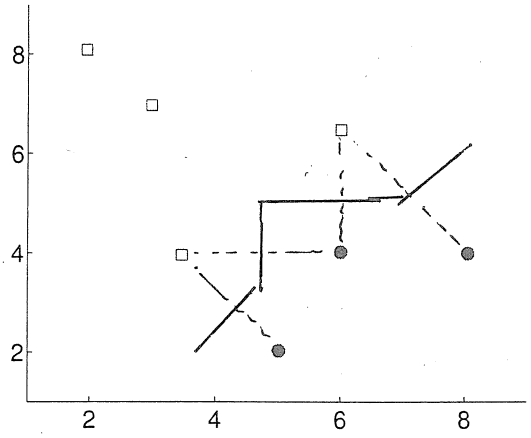
Name of the person in front of you (if any):

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.

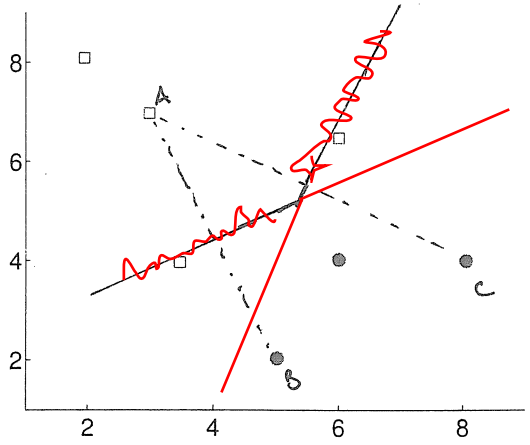
Problem 1: (9 points) K-Nearest Neighbor Classification

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).

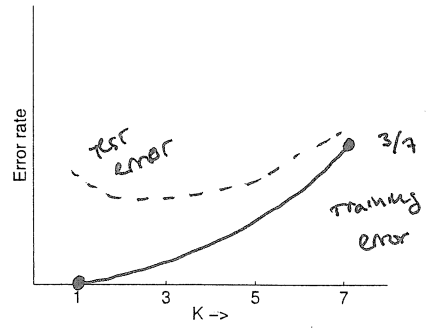
Decision boundary is piecewise linear, perpendicular & halfway along lines connecting the data points (dashed)



(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.

confusion

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.

In this weighted neighbor regression model, increasing c will indeed make it **more** likely to overfit the data. Here's why:

The weight function $w^{(i)} = \exp(-c\|x - x^{(i)}\|)$ assigns higher weights to points closer to the test point x and lower weights to those further away. When c is increased:

- The weights decay more rapidly with distance.
- As a result, only the nearest neighbors to x will significantly influence the prediction, while distant points will have minimal or no effect.

This increased focus on nearby points makes the model fit more closely to local variations in the data, increasing the likelihood of overfitting. In contrast, a smaller c would spread the

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.**

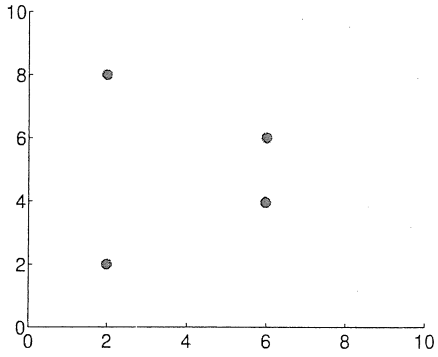
The VC dimension of our learner will most likely **increase** **stay the same** **decrease.**

(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.)

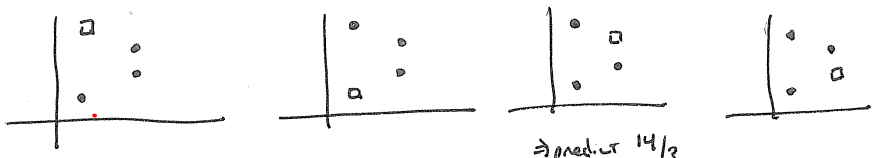
Problem 3: (8 points) Cross-validation and Linear Regression

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 = $(4)^2$

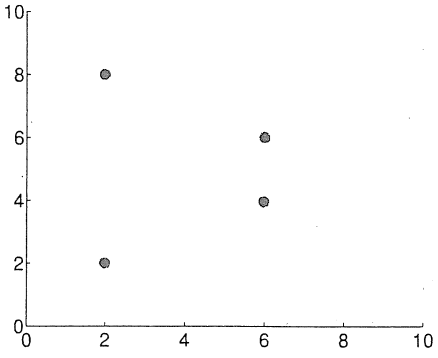
⇒ predict $14/3$ ⇒ SE = $(4/3)^2$

⇒ SE = $(4/3)^2$

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

$$= 8 + 8/9$$

(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.

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

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

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

(scalar) (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:

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

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

(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):

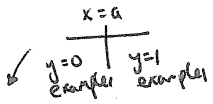
$\log_2(1) = 0$ $\log_2(2) = 1$ $\log_2(3) = 1.59$ $\log_2(4) = 2$ $\log_2(5) = 2.32$

$\log_2(6) = 2.59$ $\log_2(7) = 2.81$ $\log_2(8) = 3$ $\log_2(9) = 3.17$ $\log_2(10) = 3.32$

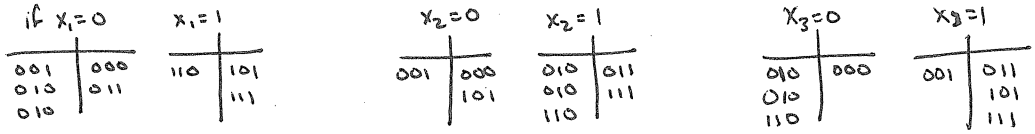
In case of ties, we prefer to use the feature with the smaller index (x_1 over x_2, etc.) and prefer to predict class 1 over class 0.

(a) What is the entropy of y?

$p(y=1) = 1/2 \Rightarrow H(y) = 1$ bit



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



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

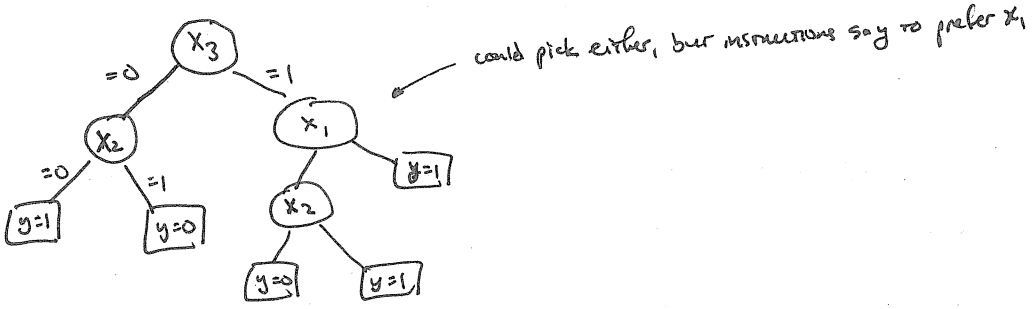
(c) What is the information gain of the variable you selected in part (b)?

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

$= 1 - 1/2\, H(3/4) - 1/2\, H(3/4)$

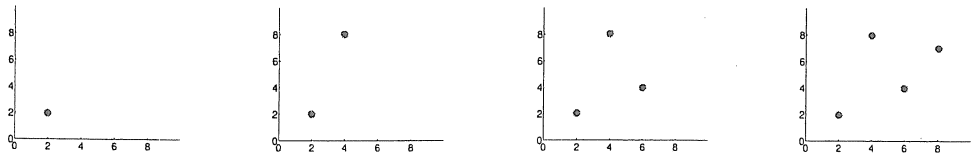
$= 1 + 3/4 \log 3/4 + 1/4 \log 1/4$

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



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.

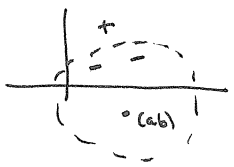
(a) $\hat{y} = T[(x_1 - a)^2 + (x_2 - b)^2 + c]$ (note: uses both x_1, x_2) – predicts +1 outside a circle at (a,b) with radius $\sqrt{-c}$

Ans: (a), (b), and (c) – but not (d).

(a) - easy

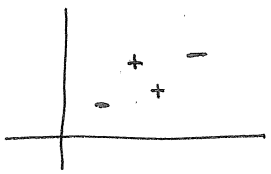
(b) - move center of the circle near "-1" datum

(c)



but

(d)



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

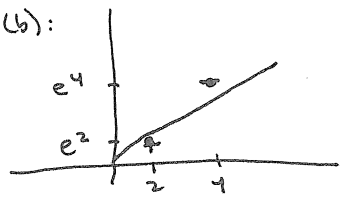
(b) $\hat{y} = T[ax_1 + b\exp(x_1)]$ (note: uses only x_1!)

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

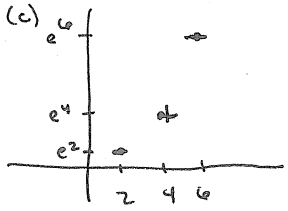
– A 2-feature perceptron w/ no intercept (bdry passes through the origin).

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

Ex (b):



Ex (c)



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

CS273 Midterm Exam

Introduction to Machine Learning: Winter 2015

Tuesday February 10th, 2014

Your name: SOLUTIONS

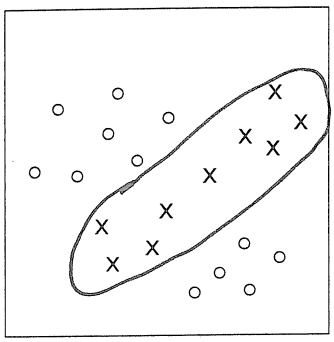
Your UCINetID (e.g., myname@uci.edu):

Your seat (row and number):

- Total time is 80 minutes. 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: (8 points) Separability and Features

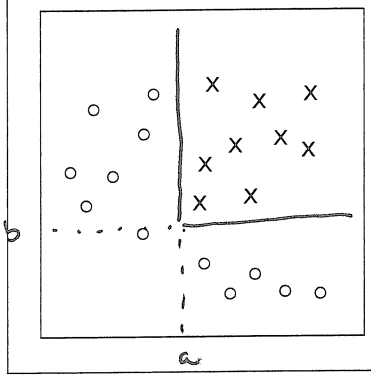
For each of the following examples of training data, **(1)** sketch a classification boundary that separates the data; **(2)** state whether or not the data are linearly separable, and if not, **(3)** give a set of features that would allow the data to be separated. (Your features do not need to be minimal, but should not contain any obviously unneeded features.)



No - not linearly separable

Could use features such as

$[1 \quad x_1 \quad x_2 \quad x_1x_2 \quad x_1^2 \quad x_2^2]$



No, not linearly separable

The same quadratic features would work, or eg.

$[1 \quad x_1 > a \quad x_2 > b]$

also works.

Problem 2: (8 points) Under- and Over-fitting

Suppose that I am training a neural network classifier to recognize faces in images. Using cross-validation, we discover that my classifier appears to be overfitting the data. Give two ways I could improve my performance – be specific.

Lots of possible answers:

Regularize
Use fewer hidden nodes
Use fewer layers
Get more data (if possible!)
Use "early stopping"
Use fewer input features / feature selection

After following some of your advice, we now think that the resulting classifier is underfitting. Give two ways, **other than** reversing the methods you mentioned above, that we could improve performance; again, be specific.

Just reverse two ideas in the other part

eg: generate new features (polynomials, etc)

or increase the # of hidden nodes.

Problem 3: (9 points) Bayes Classifiers and Naïve Bayes

Consider the table of measured data given at right. We will use the two observed features x_1, x_2 to predict the class y. In the case of a tie, we will prefer to predict class $y = 0$.

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

(a) Write down the probabilities necessary for a naïve Bayes classifier:

$p(y=1) = 5/8$

$p(x_1=1 \mid y=0) = 2/3$ $\quad p(x_1=1 \mid y=1) = 0$.

$p(x_2=1 \mid y=0) = 1/3$ $\quad p(x_2=1 \mid y=1) = 3/5$.

(b) Using your naïve Bayes model, what value of y is predicted given observation $(x_1, x_2) = (00)$?

Compare: $p(y=0)\, p(x_1=0 \mid y=0)\, p(x_2=0 \mid y=0)$ vs $p(y=1)\, p(x_1=0 \mid y=1)\, p(x_2=0 \mid y=1)$

$3/8 \cdot 1/3 \cdot 2/3$ vs $5/8 \cdot 1 \cdot 2/5$

$= \frac{6}{72}$ vs $1/4$.

$\Rightarrow$ predict $\hat{y} = 1$.

(c) Using your naïve Bayes model, what is the probability $p(y = 1 | x_1 = 0, x_2 = 1)$?

Compare: $p(y=0)\, p(x_1=0 \mid y=0)\, p(x_2=1 \mid y=0)$ $\quad p(y=1)\, p(x_1=0 \mid y=1)\, p(x_2=1 \mid y=1)$

$3/8 \cdot 1/3 \cdot 1/3$ $\quad 5/8 \quad 1 \quad 3/5$

$= \frac{3}{8 \cdot 9}$ $\quad \frac{3}{8}$

$\Rightarrow p(y=1 \mid x=01) = \frac{3/8}{3/8 + 3/72} = \frac{27}{27+3} = \frac{27}{30} = .9$.

Problem 4: (10 points) Gradient Descent

Suppose that we have training data $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$
and we wish to predict y using a nonlinear regression model with two parameters:

$$\hat{y} = a \exp(x_1 + b)$$

We decide to train our model using gradient descent on the mean squared error (MSE).

(a) Write down the expression for the MSE on our training set.

$$J(\theta) = \frac{1}{m} \sum_i (y^i - \hat{y}^i)^2 = \frac{1}{m} \sum_i [y^i - (a \exp(x^i + b))]^2$$

(b) Write down the gradient of the MSE.

$$\nabla J = \left[\frac{\partial J}{\partial a} \quad \frac{\partial J}{\partial b}\right]$$

$$\frac{\partial J}{\partial a} = \frac{1}{m} \sum_i (2) [y^i - \hat{y}^i] (-1) [\exp(x^i + b)]$$

$$\frac{\partial J}{\partial b} = \frac{1}{m} \sum_i (2) [y^i - \hat{y}^i] (-1) [a \exp(x^i + b)]$$

(c) Give pseudocode for a (batch) gradient descent function `theta = train(X,Y)`, including all necessary elements for it to work.

Initialize $\theta = [a \; b] = [\emptyset \; \emptyset]$; $\theta^{old} = \theta$.

Init step size α $(= 1)$, stopping tolerance ϵ $(= 1e^{-3})$

While (¬ done) {

$\theta \leftarrow \theta - \alpha \nabla J$ (from (b))

if $(\|\theta - \theta^{old}\|_2^2 < \epsilon)$ done = true

$\theta^{old} = \theta$.

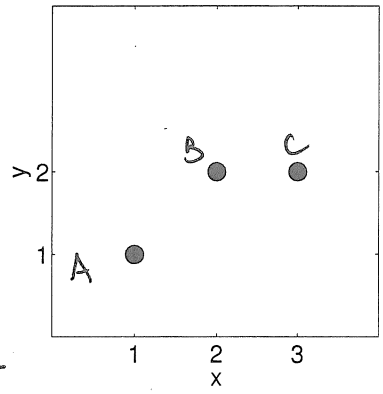
}.

Problem 5: (8 points) Cross-validation and Linear Regression

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

(a) Compute the leave-one-out cross-validation error of a zero-order (constant) predictor.

Mean of points



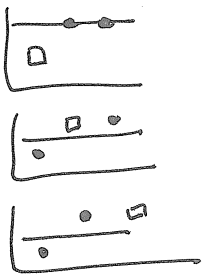
A: predict 2 $\Rightarrow$ $(1-2)^2 = 1^2$

B: predict $1\frac{1}{2}$ $\Rightarrow$ $(2-1\frac{1}{2})^2 \Rightarrow + (\frac{1}{2})^2$

C: predict $1\frac{1}{2}$ $\Rightarrow$ " $+ (\frac{1}{2})^2$

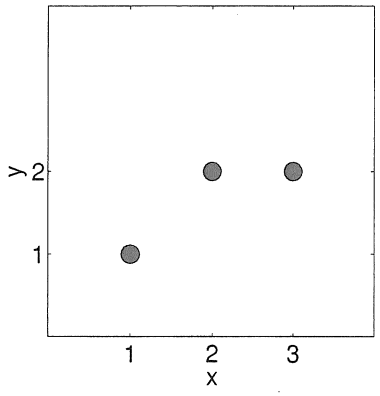
$\Rightarrow$ MSE $= \frac{1}{3}(1 + \frac{1}{4} + \frac{1}{4}) = \frac{1}{3}(1\frac{1}{2}) = \frac{1}{2}$.

$= \frac{1}{2}$.



(b) Compute the leave-one-out cross-validation error of a first-order (linear) predictor.

goes thru the points

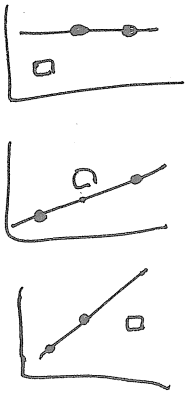


A: predict 2 $\Rightarrow$ 1^2

B: predict $1\frac{1}{2}$ $\Rightarrow$ $(\frac{1}{2})^2$

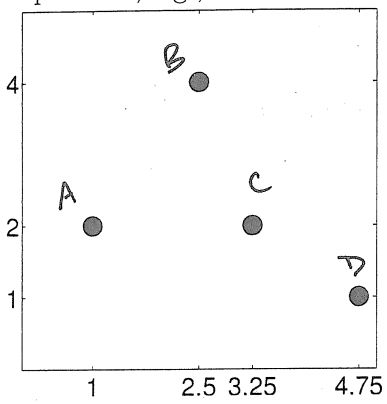
C: predict 3 $\Rightarrow$ 1^2

$\Rightarrow$ MSE $= \frac{1}{3}[1 + \frac{1}{4} + 1] = \frac{1}{3} \cdot \frac{9}{4} = \frac{3}{4}$.



Problem 6: (12 points) K-Nearest Neighbor Regression

Consider a regression problem for predicting the following data points, using the k-nearest neighbor regression algorithm to minimize mean squared error (MSE). In the case of ties, we will prefer to use the neighbor to the left (smaller x value). Note: if you prefer, you may leave an arithmetic expression, e.g., leave values as "$(.6)^2$".



(a) For $k = 1$, compute the training error on the provided data. confused

$\emptyset$

(b) For $k = 1$, compute the leave-one-out cross-validation error on the data. confused

A's NN is B $\Rightarrow$ 2^2
B's NN is C $\Rightarrow$ 2^2
C's NN is B $\Rightarrow$ 2^2
D's NN is C $\Rightarrow$ 1^2

$\Rightarrow$ $13/4$.

(c) For $k = 3$, compute the training error on the provided data.

A's 3NN : ABC
B's 3NN : ABC
C's 3NN : BCD
D's 3NN : BCD

ABC $\rightarrow$ predict $8/3$
BCD $\Rightarrow$ predict $7/3$

$\Rightarrow (2-8/3)^2 + (4-8/3)^2 + (2-7/3)^2 + (1-7/3)^2$

$= (2/3)^2 + (4/3)^2 + (1/3)^2 + (4/3)^2 = 37/9$.

(d) For $k = 3$, compute the leave-one-out cross-validation error on the data.

A's 3NN : BCD
B's 3NN : ACD $\Rightarrow$
⋮

predict $7/3$
$5/3$
$7/3$
$8/3$

$\Rightarrow (2-7/3)^2 + (4-5/3)^2 + (2-7/3)^2 + (1-8/3)^2$

$= (1/3)^2 + (7/3)^2 + (1/3)^2 + (2/3)^2$

$6 = 55/9$

Problem 7: (4 points) Multiple Choice

For the following questions, assume that we have m data points $y^{(i)}$, $x^{(i)}$, $i = 1 \ldots m$, each with n features, $x^{(i)} = [x_1^{(i)} \ \ldots \ x_n^{(i)}]$.

Circle one answer for each:

True or **false**: Linear regression can be solved using either matrix algebra or gradient descent.

True or **false**: The predictions of a k-nearest neighbor classifier will not be affected if we pre-process the data to normalize the magnitude of each feature. Changing feature scale ⇒ changes distances.

True or **false**: With enough hidden nodes, a Neural Network can separate any data set.

True or **false**: Increasing the regularization of a linear regression model will decrease the bias.

Problem 8: (4 points) Short Answer

Give one advantage of stochastic gradient descent over batch gradient descent, **and** one advantage of batch gradient descent over stochastic.

SGD: often faster, esp. (initially?) for very large data sets

Batch: less random:
easier to guage convergence, keep track of current loss value,
small step size ensures monotonic decrease in the loss.

Problem 9: (12 points) Perceptrons and VC Dimension

In this problem, consider the following perceptron model on two features:

$$\hat{y}(x) = \text{sign}(\, b + w_1 x_1 + w_2 x_2 \,)$$

and answer the following questions about the decision boundary and the VC dimension.

(a) Describe (in words, with diagrams if desired) the possible decision boundaries that can be realized by this classifier

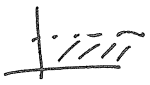
A standard perceptron – so - lines in 2D space; either decision on either side.

(b) What is its VC dimension?

3 (standard result from class)

Now suppose that I also enforce an additional condition on the parameters of the model: that **only one** of the two weights w_1, w_2 is non-zero (i.e., one of them must be zero, a "feature selection" criterion). Note that the training algorithm can choose which parameter is zero, depending on the data.

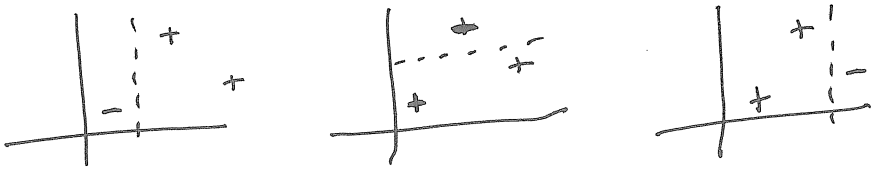
(c) Describe (in words, with diagrams if desired) the decision boundaries that can be realized by this classifier (there should be two "cases").

if $w_1 = 0 \Rightarrow$  horizontal boundaries (± 1 either side)

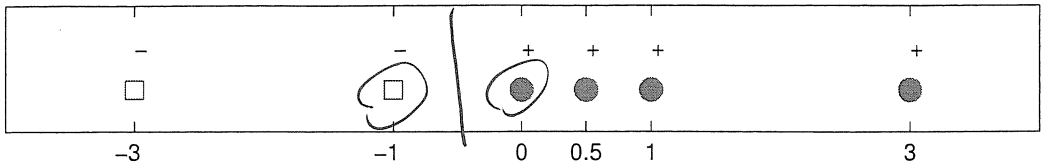
if $w_2 = 0 \Rightarrow$  vertical boundaries (± 1 either side)

(d) What is its VC dimension?

Still 3:



Problem 10: (9 points) Support Vector Machines



Using the above data with one feature x (whose values are given below each data point) and a class variable $y \in \{-1, +1\}$, with filled circles indicating $y = +1$ and squares $y = -1$ (the sign is also shown above each data point for redundancy), answer the following:

(a) Sketch the solution (decision boundary) of a linear SVM on the data, and identify the support vectors. Boundary at $x = -1/2$ SVs are $x=-1$, $x=0$.

(b) Give the solution parameters w and b, where the linear form is $wx + b$.

$w(-1) + b = -1$

$w(-1/2) + b = 0$. $\Rightarrow$ $b = +1$

$w(0) + b = +1$ $w = 2$.

(c) Calculate the training error:

ϕ - separates the data.

(d) Calculate the leave-one-out cross-validation error for these data:

If any points except $x=-1$, $x=0$ are left out, the boundary stays the same $\Rightarrow$ correct.

If $x=1$ left out, boundary moves to -1.5 $\Rightarrow$ X

$x=0$ left out, " " to $-.25$ $\Rightarrow$ ✓

$\Rightarrow$ $1/6$

CS273a Midterm Exam

Introduction to Machine Learning: Fall 2016

Tuesday November 1st, 2016

Your name: SOLUTIONS

Your ID # and UCINetID (e.g., 123456789, myname@uci.edu):

Your seat (row and number):

- Total time is 80 minutes. 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 or TA to come over.
- Turn in any scratch paper with your exam

(This page intentionally left blank)

Problem 1: (10 points) Bayes Classifiers

In this problem you will use Bayes Rule: $p(y|x) = p(x|y)p(y)/p(x)$ to perform classification. Suppose we observe some training data with two binary features x_1, x_2 and a binary class y. After learning the model, you are also given some validation data.

Table 1: Training Data

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

Table 2: Validation Data

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

In the case of any ties, we will prefer to predict class 0.

(a) Give the predictions of a joint Bayes classifier on the validation data. What is the validation error rate?

$\hat{y}$:

00 ⇒ 0 ×
01 ⇒ 1 ×
10 ⇒ 1 ✓
11 ⇒ 0 ✓

⇒ validation error rate = 1/2.

(b) Give the predictions of a naïve Bayes classifier on the validation data. What is the validation error rate?

$p(y=0) = p(y=1) = 1/2$

$p(x_1=1 \mid y=0) = 1/2$ $\quad p(x_1=1|y=1) = 1/2$

$p(x_2=1 \mid y=0) = 3/4$ $\quad p(x_2=1|y=1) = 1/2$

⇒

$\hat{y}$:

00 ⇒ 1
01 ⇒ 0
10 ⇒ 1
11 ⇒ 0

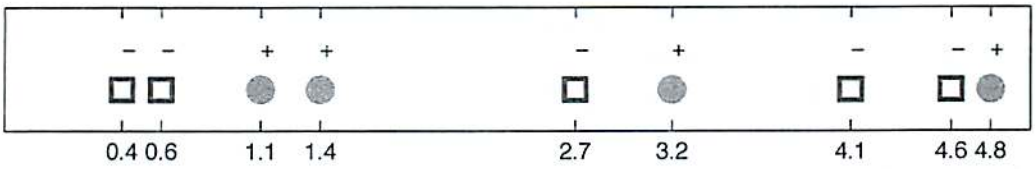
⇒ validation error rate = ∅.

(c) **True** or **False**: In a naïve Bayes model, the features x_i are independent, i.e., $p(x_1, x_2) = p(x_1)p(x_2)$.

The features are conditionally independent, $p(x_1, x_2|y) = p(x_1|y)\,p(x_2|y)$

but not independent.

Problem 2: (9 points) Nearest Neighbor Classification



Given the above data with one scalar feature x (whose values are given below each data point) and a class variable $y \in \{-1, +1\}$, with filled circles indicating $y = +1$ and squares $y = -1$ (the sign is also shown above each data point for redundancy), we use a k-nearest neighbor classifier to perform prediction; in the case of ties, we prefer to predict class -1. Answer the following:

(a) Compute the training error rate of a 1-Nearest-Neighbor classifier trained on these data.

ϕ

bc each point includes itself while calculating the smallest distance

(b) Compute the leave-one-out cross-validation error rate of a 1-Nearest-Neighbor classifier on these data.

check nearest non-self point.

✓✓ ✓✓ ✗ ✗ ✓ ✗ ✗

$\Rightarrow 4/9$

(c) Compute the training error for a 3-Nearest-Neighbor classifier on these data.

check nearest 3 points, including self.

✓✓ ✓✓ ✗ ✗ ✓✓ ✗

$\Rightarrow 3/9$.

Problem 3: (10 points) Gradient Descent

Suppose that we have training data $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$, where $x^{(i)}$ is a scalar feature and $y^{(i)} \in \{-1, +1\}$, and we wish to train a linear classifier, $\hat{y} = \text{sign}[a + bx]$, with two parameters a, b. In order to train the model, we use gradient descent on a smooth surrogate loss called the *exponential loss*:

$$J(X, Y) = \frac{1}{m} \sum_i \exp\left(y^{(i)}(a + bx^{(i)})\right)$$

NOTE - this loss has a typo; it should be $\exp(-y^{(i)}(a+bx^{(i)}))$. We'll take the gradient of the loss as given here.

(a) Write down the gradient of our surrogate loss function.

$$\nabla J = \left[\frac{\partial J}{\partial a} \quad \frac{\partial J}{\partial b}\right]$$

$$\frac{\partial J}{\partial a} = \frac{1}{m} \sum_i \exp\left[y^{(i)}(a + bx^{(i)})\right] \cdot y^{(i)}$$

$$\frac{\partial J}{\partial b} = \frac{1}{m} \sum_i \exp\left[y^{(i)}(a + bx^{(i)})\right] y^{(i)} x^{(i)}.$$

(b) Give one advantage of batch gradient descent over stochastic gradient.

Easier to test for convergence

Easy to make monotonic

⋮

(c) Give pseudocode for a (batch) gradient descent function `theta = train(X,Y)`, including all necessary elements for it to work.

Init $\theta = [a, b]$ (random, zero, etc).

Select step size α

while (¬done) {

  Compute ∇J as in part (a)

  $\theta \leftarrow \theta - \alpha \nabla J$

  check if done, eg: $\|\nabla J\| < \epsilon$; # of iterations $> T$; etc.

}

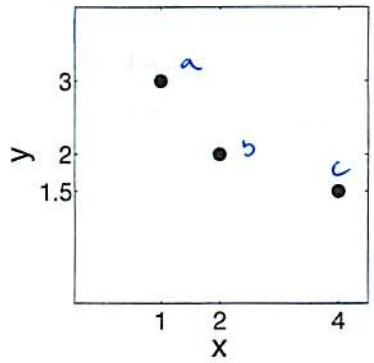
return θ.

Problem 4: (10 points) Linear Regression, Cross-validation

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

(a) Compute the **leave-one-out** cross-validation error of a zero-order (constant) predictor,

$$\hat{y}(x) = \theta_0$$



Best constant predictor: mean of the training data.

Leave out:

a ⇒ $\theta_0 = 1.75$

b $= 2.25$ ⇒

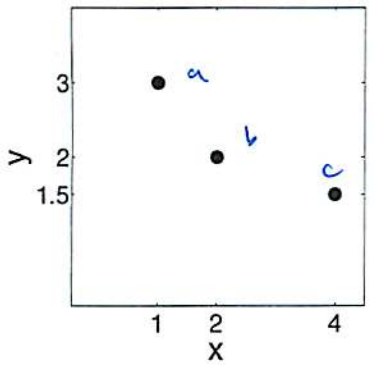
c $= 2.5$

$MSE = (1.25)^2 + (.25)^2 + (1)^2$

⇒ Xval MSE = $\frac{1}{3}\left[(5/4)^2 + (1/4)^2 + (1)^2\right] = \frac{1}{3}\left(\frac{42}{16}\right) = 7/8$.

(b) Compute the **leave-one-out** cross-validation error of a first-order (linear) predictor,

$$\hat{y}(x) = \theta_0 + \theta_1 x$$



Leave one out ⇒ best line for remaining two points:

a ⇒ $\hat{y} = 2.25$

b ⇒ $= 2.5$ ⇒

c ⇒ $= \emptyset$

$MSE = (.75)^2 + (.5)^2 + (1.5)^2$

Xval MSE = $\frac{1}{3}\left((3/4)^2 + (2/4)^2 + (6/4)^2\right)$

$= 49/48$.

Problem 5: (20 points) Multiple Choice

For the following questions, assume that we have m data points $y^{(i)}$, $x^{(i)}$, $i = 1 \ldots m$, each with n features, $x^{(i)} = [x_1^{(i)} \ \ldots \ x_n^{(i)}]$.

Circle one answer for each:

Suppose that we are training a linear classifier (perceptron). Before training, we decide to remove (throw away) 10% of our features (selected at random). This is most likely to make it **more** **equally** **less** likely to overfit the data.

When training a k-nearest neighbor model, we decide to increase the value of k. This will most likely make our model **more** **equally** **less** likely to overfit the data.

Again, training a k-nearest neighbor model, we double the amount of data available to the model. We then re-train the model, including re-optimizing k.
This is likely to **increase** **not change** **decrease** the bias.

Suppose that, when training a linear regressor, we double the amount of data available for training. This is most likely to decrease the **bias** **variance** **both** **neither** of our learned model.

Still training a linear regressor, instead of providing more real data, we instead include m additional points of "fake" data, $(x^{(i)}, y^{(i)}) = (0, 0)$.
This will most likely **increase** **not change** **decrease** the bias.
It will most likely **increase** **not change** **decrease** the variance.

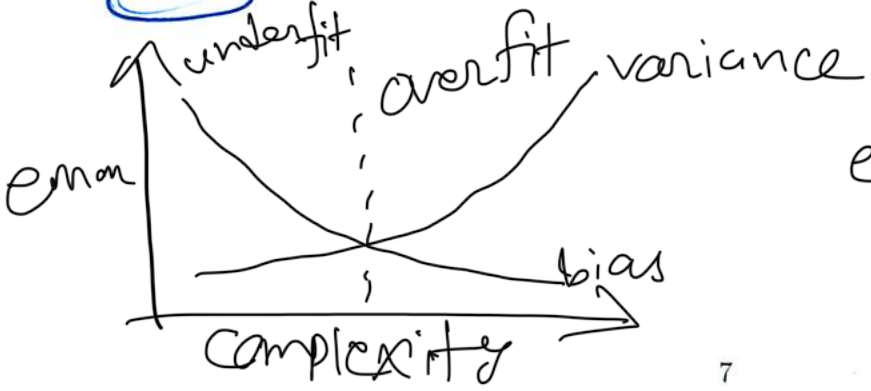
True or **false**: if the VC dimension of a model is H, then the model can shatter any set of H training points.

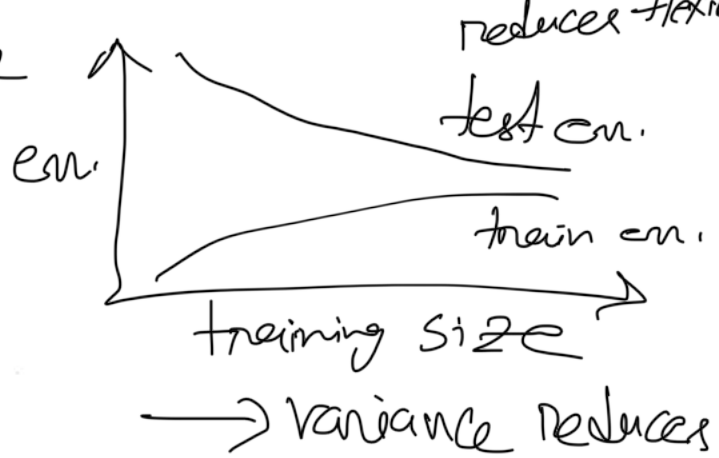
True or **false**: Linear regression can be solved using either matrix algebra or gradient descent.

True or **false**: Increasing the regularization of a linear regression model will decrease the variance.

Before training a linear classifier, we transform one of our features by taking its logarithm, i.e., `X[:,1] = np.log(X[:,1]);`. This is likely to **increase** **not change** **decrease** the model's VC dimension.

We train a Gaussian Bayes classifier, but then decide to re-train it, forcing the two classes' covariance matrices to be equal, i.e., $\Sigma_{(y=+1)} = \Sigma_{(y=-1)}$. This is likely to **increase** **not change** **decrease** the variance of our model.





Problem 6: (9 points) Short answer

Consider the two possible decision boundaries (indicated by Line 1 and Line 2) for the binary classification problem shown in Figure 1. For each algorithm below, will it possibly produce boundary 1, boundary 2, or both? Please give a concise explanation of your choice.

Perceptron Algorithm :

1 or 2 - both lines separate the data & so the perceptron algorithm will not update them.

Logistic Regression :

1 only - assigns higher likelihood to the data than L2; objective is convex (no local optima), so if reasonably optimized it will not produce L2.

Support Vector Machine (hard-margin) :

1 only - maximum margin.

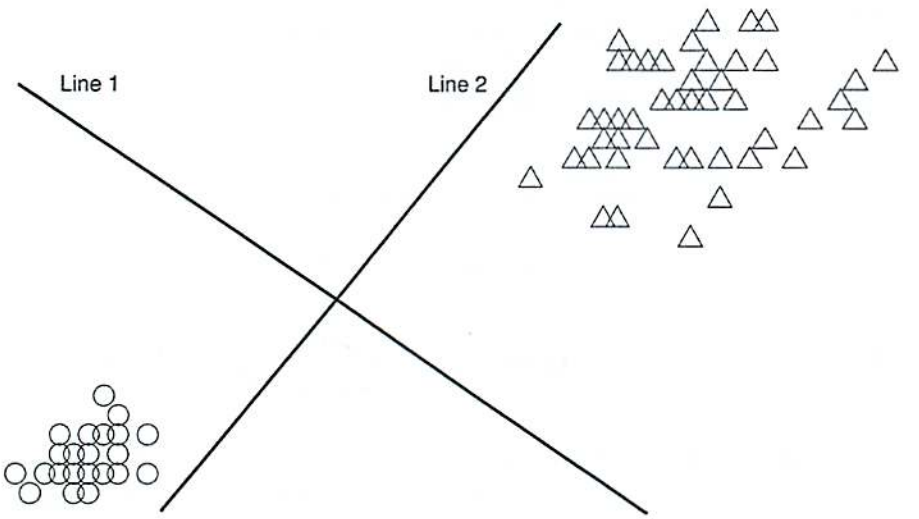


Figure 1: Possible linear decision boundaries.

Problem 7: (10 points) Support Vector Machines

Suppose we are learning a linear support vector machine with a single scalar feature x and binary target $y \in \{-1, +1\}$. We observe training data:

$$D = \{(x^{(i)}, y^{(i)})\} = \{(0, +1), (-3, +1), (1, -1)\}$$

Our linear classifier takes the form $f(x; a, b) = \text{sign}(ax + b)$.

(a) Write down the primal optimization problem for a support vector machine on these data.

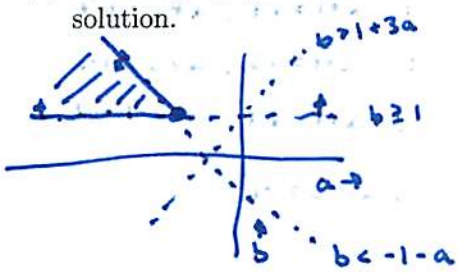
$\min a^2$

s.t. $a \cdot 0 + b \geq +1$

$a \cdot -3 + b \geq +1$

$a \cdot 1 + b \leq -1$

(b) Sketch (graph) the constraint set on the parameters a, b, and give the values of a, b at the solution.

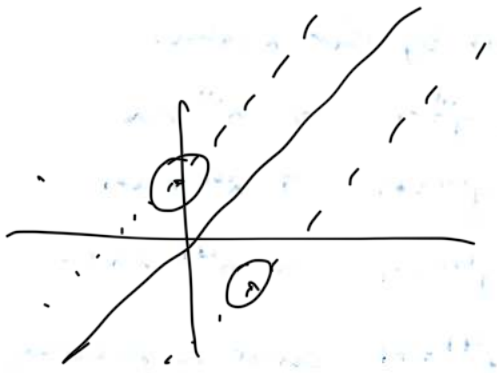


Solution is $b = 1$

$a + b = -1 \Rightarrow a = -2$

(c) Identify the support vectors.

SVs: (0, +1) and (1, −1)



(d) Give two possible advantages of the *dual* form of the SVM over the primal.

Easy to initialize to a valid (feasible) point

Can use kernel similarity f'n, equivalent to large or infinite # of features

May be more efficient if n small and n is large (# data) (# features)

computationally efficient, requires fewer variables since most α_i's = 0

Problem 8: (10 points) VC Dimension

Consider the following classifier, parameterized by a single scalar parameter a and operating on a scalar feature x:

$$f(x\ ;\ a) = \begin{cases} +1 & x \le a \text{ or } a+1 < x \le a+2 \\ -1 & \text{otherwise} \end{cases}$$

In this problem, we will show the VC dimension of $f(x\ ;\ a)$ is 3.

(a) Show by example that $f(x\ ;\ a)$ can shatter three points. Hint: place your points at $x^{(1)} = 0$, $x^{(2)} = 0.75$, $x^{(3)} = 1.5$.

Check each pattern:

+ ++	$\Rightarrow$	$a > 1.5$
+ + −	$\Rightarrow$	$a = 1$
+ − +	$\Rightarrow$	$a = 0.25$
+ − −	$\Rightarrow$	$a = 0.7$
− + +	$\Rightarrow$	$a = -0.3$: $a+1 = .7$, $a+2 = 1.7$
− + −	$\Rightarrow$	$a = -0.7$: $a+1 = .3$, $a+2 = 1.3$
− − +	$\Rightarrow$	$a = -0.1$: $a+1 = .9$ $a+2 = 1.9$
− − −	$\Rightarrow$	$a = -3$: $a+2 = -1$

(b) Argue that $f(x\ ;\ a)$ cannot shatter four points. (Which target pattern cannot be reproduced?)

It cannot ~~shatter~~ produce the pattern − + − + $\Rightarrow$ cannot shatter 4 points.

Order the points by their x value.

1st point, $y^{(1)} = -1$ $\Rightarrow$ $x^{(1)} > a$

2nd $y^{(2)} = +1$ $\Rightarrow$ $x^{(2)} > a+1$

$y^{(3)} = -1$ $\Rightarrow$ $x^{(3)} > a+2$

But then $y^{(4)} = +1$ cannot be predicted, because $x^{(4)} > x^{(3)} > a+2$

$\Rightarrow \hat{y}(x^{(4)}) = -1$.

CS273A Midterm Exam

Introduction to Machine Learning: Winter 2019

Tuesday February 12th, 2019

Your name: SOLUTIONS

Row/Seat Number: Lecture

Your ID #(e.g., 123456789) 31415926S

UCINetID (e.g.ucinetid@uci.edu) pattearc@uci.edu

- Please put your name and ID **on every page**.
- Total time is 80 minutes. 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 or TA to come over.
- You may use **one** sheet containing handwritten notes for reference, and a (basic) calculator.
- Turn in your notes and any scratch paper with your exam.

Problems

Total, *(64 points.)*

Name: _____ ID#: _____

Bayes Classifiers, *(10 points.)*

x_1	x_2	y
a	b	0
b	c	0
b	c	0
c	c	0
a	c	1
a	b	1
b	a	1
b	b	1

Consider the table of measured data given at right. We will use the two observed features x_1, x_2 to predict the class y. Each feature can take on one of three values, $x_i \in \{a, b, c\}$.
In the case of a tie, we will prefer to predict class $y = 0$.

(1) Write down the probabilities learned by a naïve Bayes classifier: *(4 points.)*

$p(y=0)$: $1/2$ $\quad$ $p(y=1)$: $1/2$

$p(x_1=a\,|\,y=0)$: $1/4$ $\quad$ $p(x_1=a\,|\,y=1)$: $1/2$

$p(x_1=b\,|\,y=0)$: $1/2$ $\quad$ $p(x_1=b\,|\,y=1)$: $1/2$

$p(x_1=c\,|\,y=0)$: $1/4$ $\quad$ $p(x_1=c\,|\,y=1)$: 0

$p(x_2=a\,|\,y=0)$: 0 $\quad$ $p(x_2=a\,|\,y=1)$: $1/4$

$p(x_2=b\,|\,y=0)$: $1/4$ $\quad$ $p(x_2=b\,|\,y=1)$: $1/2$

$p(x_2=c\,|\,y=0)$: $3/4$ $\quad$ $p(x_2=c\,|\,y=1)$: $1/4$.

(2) Using your naïve Bayes model, compute: *(3 points.)*
$p(y=0|x_1=a, x_2=c)$: $3/4$ $\quad$ $p(y=1|x_1=a, x_2=c)$: $1/4$

$= \dfrac{1/2 \cdot 1/4 \cdot 3/4}{1/2 \cdot 1/4 \cdot 3/4 + 1/2 \cdot 1/2 \cdot 1/4} = \dfrac{3}{3+1} = 3/4$ $\quad$ $= \dfrac{1/2 \cdot 1/2 \cdot 1/4}{1/2 \cdot 1/4 \cdot 3/4 + 1/2 \cdot 1/2 \cdot 1/4} = 1/4.$

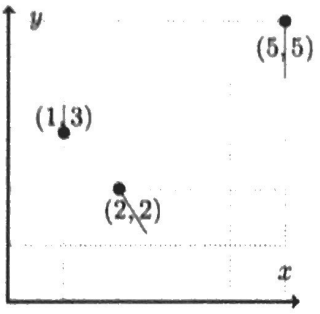
(3) Compute the probabilities $p(y=0|x_1=a, x_2=c)$ and $p(y=1|x_1=a, x_2=c)$ for a **joint** (not naïve) Bayes model trained on the same data. *(3 points.)*

$p(y=0|a,c) = 0$ $\quad$ $p(y=1|a,c) = 1$.

Name: ID#:

Linear and Nearest Neighbor Regression, *(12 points.)*

Consider the data points shown at right, for a regression problem to predict y given a scalar feature x.



(1) Compute **training** MSE of a 1-nearest neighbor predictor. *(3 points.)*

0.

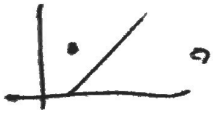
(2) Compute the **leave-one-out** cross-validation error (MSE) of a 1-nearest neighbor predictor. *(3 points.)*

$\frac{1}{3}\left[1^2 + 1^2 + 3^2\right] = 11/3.$

(3) Compute the **leave-one-out** cross-validation error (MSE) of a 2-nearest neighbor predictor. *(3 points.)*

$1/3\left[1/2^2 + 2^2 + 2\frac{1}{2}^2\right] = 7/2.$

(4) Compute the **leave-one-out** cross-validation error (MSE) of a linear regressor, e.g., a model of the form $f(x) = \theta_0 + \theta_1 x$. *(3 points.)*


$\Rightarrow 2^2$

$\Rightarrow \frac{1}{3}\left[2^2 + 1\frac{1}{2}^2 + 6^2\right] = \frac{457}{12} = 38\frac{1}{12}.$


$\Rightarrow (1\frac{1}{2})^2$


$\Rightarrow 6^2$

Name: _____ ID#: _____

Multiple Choice, *(12 points.)*

Here, assume that we have m data points $y^{(i)}, x^{(i)}$, $i = 1 \ldots m$, each with n features, $x^{(i)} = [x_1^{(i)} \ \ldots \ x_n^{(i)}]$. For each of the choices below, will it likely increase, decrease, or have no effect on overfitting (circle your choice)? If you think it is equally likely to go either way, pick *No Effect*.

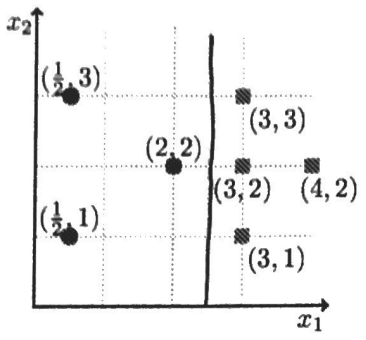
1	Gathering more labeled training data	**Reduce**	**Increase**	**No Effect**
2	For a linear regressor, use $2 \times m$ training data by adding m all-zero (x and y) data points.	**Reduce**	**Increase**	**No Effect**
3	For a linear regressor, use $2 \times n$ features per data point by adding n random values to each.	**Reduce**	**Increase**	**No Effect**
4	For a linear regressor, use $2 \times n$ features per data point by adding n all-zero features to each.	**Reduce**	**Increase**	**No Effect**
5	For a linear regressor, increasing the $L2$ regularization penalty	**Reduce**	**Increase**	**No Effect**
6	For a 3-nearest neighbor classifier, use $2 \times m$ training data by copying (duplicating) each data point.	**Reduce**	**Increase**	**No Effect**
7	For a 3-nearest neighbor classifier, use $2 \times n$ features per data point by copying (duplicating) the features.	**Reduce**	**Increase**	**No Effect**
8	For a k-nearest neighbor classifier, rescaling the data to zero mean, unit variance	**Reduce**	**Increase**	**No Effect**
9	For a neural network model, increasing the number of hidden nodes in the first layer	**Reduce**	**Increase**	**No Effect**
10	For a neural network, changing the activation function of the hidden nodes from logistic (sigmoid) to rectified linear (ReLU).	**Reduce**	**Increase**	**No Effect**
11	Switching from linear to polynomial Kernel SVMs	**Reduce**	**Increase**	**No Effect**
12	Using gradient descent to optimize our SVM model, rather than a QP (quadratic program) solver.	**Reduce**	**Increase**	**No Effect**

Name: _____ ID#: _____

Support Vector Machines, *(10 points.)*

Suppose we are learning a linear support vector machine with two real-valued features x_1, x_2 and binary target $y \in \{-1, +1\}$. We observe training data (pictured at right):

x_1	x_2	y
0.5	1	-1
2	2	-1
0.5	3	-1
3	2	+1
3	1	+1
3	3	+1
4	2	+1



Our linear classifier takes the form

$$f(x; w_1, w_2, b) = \text{sign}(w_1 x_1 + w_2 x_2 + b).$$

(1) Consider the optimal linear SVM classifier for the data, i.e., the one that separates the data and has the largest margin. **Sketch** its decision boundary in the above figure, and **list** the support vectors here. *(2 points.)* (2,2) ; (3,3), (3,2), (3,1).

(2) Derive the parameter values w_1, w_2, b of this $f(x)$ using these support vectors. What is the length of the margin? *(3 points.)*

$w_1 \cdot 2\frac{1}{2} + w_2 \cdot (x_2) + b = \phi$. $\Rightarrow$ $w_2 = \phi$.

$w_1 \cdot 2 + b = -1$

$w_1 \cdot 3 + b = +1$ $\Rightarrow$ $w_1 = 2$

$b = -3$.

$M = \frac{2}{\sqrt{w_1^2 + w_2^2}} = 1$

(or by inspection).

(3) What is the *training error* of a linear SVM on these data? *(2 points.)*

ϕ.

(4) What is the the *leave-one-out cross validation error* for a linear SVM trained on these data? *(3 points.)*

1/7

point (2,2) left out $\Rightarrow$ boundary shifts & misclassifies.

other - boundary is stable.

Name: _____ ID#: _____

Gradient Descent, ***(10 points.)***

Suppose that we have training data $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$, where $x^{(i)}$ is a scalar feature and $y^{(i)} \in \{-1, +1\}$, and we wish to train a linear classifier, $\hat{y} = \text{sign}[a + bx]$, with two parameters a, b. In order to train the model, we decide to use gradient descent on a smooth surrogate loss called the *exponential loss*:

$$J(X, Y) = \frac{1}{m} \sum_{i=1}^{m} \exp\left(-y^{(i)}(a + bx^{(i)}) \right) \qquad (*)$$

(1) Write down the gradient of our surrogate loss function.

$$\frac{\partial J}{\partial a} = \frac{1}{m} \sum_i \exp\left(-y^i (a + bx^i)\right) \cdot (-y^i)$$

$$\frac{\partial J}{\partial b} = \frac{1}{m} \sum_i \exp\left(-y^i (a + bx^i)\right) \cdot (-y^i x^i) \qquad (**)$$

$$\nabla J = \left[\frac{\partial J}{\partial a} \quad \frac{\partial J}{\partial b} \right].$$

(2) Give one advantage of batch gradient descent over stochastic gradient

Easier to monitor convergence

Monotonic descent

(Also a easier to set step size schedule, etc) & more...

(3) Give one advantage of stochastic gradient descent over batch gradient

Faster for large data sets (m), particularly early in optimization

Often avoids shallow local minima

⋮

(4) Give pseudocode for a (batch) gradient descent function `theta = train(X,Y)`, including all necessary elements for it to work.

```
Initialize θ to something (zero, random, etc).
Set step size α, stopping tolerance ε
Init J^old = ∞, J = ∞.
while (|J - J^old| > ε) {        // or some other stopping criterion
    θ ← θ - α ∇J.                // ∇J in (**).
    J^old = J
    J = 1/n Σ ...                // (*) def of J.
```

Name: _____ ID#: _____

VC-Dimensionality, *(10 points.)*

Consider the VC dimension of two classifiers defined using two features x_1, x_2.

(1) First, consider a simple classifier f_A that predicts class +1 within a ring with inner radius r and a width of w:

$$f_A(x) = \begin{cases} +1 & (r < (x_1^2 + x_2^2) < r + w) \\ -1 & \text{otherwise} \end{cases}$$

Show that this classifier has VC dimension 2. *(5 points.)*

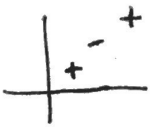
It only matters what each point's radius, $r^i = (x_1^i)^2 + (x_2^i)^2$ is; our two (or 3) points will be located at $(x_1^1)^2+(x_2^1)^2 = r^1 < r^2 = (x_1^2)^2 + (x_2^2)^2 < r^3 = (x_1^3)^2+(x_2^3)^2$.

2 points:

r<0, r'<w<r²

r'<r<r²<r+w.

3 points: the learner cannot shatter b/c:



$r < r^1 < r+w < r^2$

but then point at r^2 is mis-predicted.

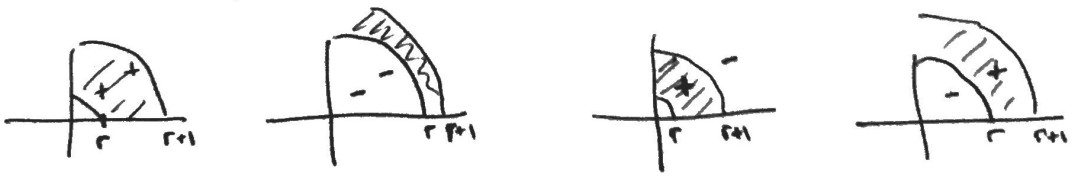
(2) Now, suppose that we fix $w = 1$, i.e., it is no longer a parameter of the model:

$$f_B(x) = \begin{cases} +1 & (r < (x_1^2 + x_2^2) < r + 1) \\ -1 & \text{otherwise} \end{cases}$$

What is the VC dimension of f_B? Justify your answer. *(5 points.)*

It turns out, this does not change the VC dimension (still 2).

Place the points so that $r^2 - r^1 < 1$. Then:



(not to scale) ☺

CS273A Midterm Exam

Introduction to Machine Learning: Winter 2020

Tuesday February 11th, 2020

Your name: Solutions

Row/Seat Number:

Your ID #(e.g., 123456789) 31415926S

UCINetID (e.g.ucinetid@uci.edu) pantcater@uci.

- Please put your name and ID on every page.
- Total time is 80 minutes. 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 or TA to come over.
- You may use one sheet containing handwritten notes for reference, and a (basic) calculator.
- Turn in your notes and any scratch paper with your exam.

Problems

Total, *(64 points.)*

Name: ID#:

True/False, (12 points.)

Here, assume that we have m data points $y^{(i)}$, $x^{(i)}$, $i = 1 \ldots m$, each with n features, $x^{(i)} = [x_1^{(i)} \ \ldots \ x_n^{(i)}]$. For each of the scenarios below, circle one of "true" or "false" to indicate whether you agree with the statement.

True or **false**: In a soft-margin SVM (i.e., loss $\sum_j w_j^2 + R \sum_i \epsilon^{(i)}$), increasing the value of R will make the model more likely to overfit.

True or **false**: A soft-margin SVM model is harder to optimize than a hard-margin SVM, since it is not a quadratic program.

True or **false**: A kernel SVM will be more efficient than a linear SVM when the number of training data, m, is large.

True or **false**: Applying "early stopping" by increasing the convergence tolerance in SGD increases the bias of the learner to reduce overfitting.

True or **false**: When training a perceptron using the logistic negative log-likelihood loss, gradient descent can never become stuck in a local optimum. convex loss fn

True or **false**: Given sufficently many data m, the 1-nearest neighbor classifier error rate approaches the Bayes optimal error rate. 2x Bayes optimal

True or **false**: Stochastic gradient descent is often preferred over batch when the number of data points m is very large.

True or **false**: For a perceptron, increasing the regularization penalty of a linear regression model will decrease the resulting model's variance.

True or **false**: For a perceptron, doubling the number of training data available will decrease the resulting model's bias.

True or **false**: For a perceptron, using $2 \times n$ features per data point by adding n random values to each will increase the resulting model's variance.

True or **false**: With enough hidden nodes, a neural network can approximate any function.

True or **false**: Using backpropagation to train a neural network will avoid getting stuck in local optima.

Name: _____ ID#: _____

Bayes Classifiers, *(10 points.)*

Consider the table of measured data given at right. We will use the two observed features x_1, x_2 to predict the class y. Each feature can take on one of three values, $x_i \in \{a, b, c\}$. In the case of a tie, we will prefer to predict class $y = 0$.

x_1	x_2	y
c	b	0
b	b	0
b	c	0
a	c	1
a	c	1
a	b	1
a	a	1
b	b	1
c	a	1

(1) Write down the probabilities learned by a naïve Bayes classifier: *(4 points.)*

$p(y=0)$: $1/3$ $\quad$ $p(y=1)$: $2/3$

$p(x_1=a\,|\,y=0)$: ϕ $\quad$ $p(x_1=a\,|\,y=1)$: $2/3$

$p(x_1=b\,|\,y=0)$: $2/3$ $\quad$ $p(x_1=b\,|\,y=1)$: $1/6$

$p(x_1=c\,|\,y=0)$: $1/3$ $\quad$ $p(x_1=c\,|\,y=1)$: $1/6$

$p(x_2=a\,|\,y=0)$: ϕ $\quad$ $p(x_2=a\,|\,y=1)$: $1/3$

$p(x_2=b\,|\,y=0)$: $2/3$ $\quad$ $p(x_2=b\,|\,y=1)$: $1/3$

$p(x_2=c\,|\,y=0)$: $1/3$ $\quad$ $p(x_2=c\,|\,y=1)$: $1/3$

(2) Using your naïve Bayes model, what value of y would you predict given $(x_1 = a, x_2 = b)$?: *(3 points.)*

$\hat{y}=1$ $\quad$ $p(x=ab\,|\,y=0)=\phi$.

(3) Using your naïve Bayes model, compute the probabilities: *(3 points.)*

$p(y=0|x_1=b, x_2=c)$: $\quad$ $p(y=1|x_1=b, x_2=c)$: $1/3$

$$= \frac{1/2 \cdot 2/3 \cdot 1/3}{1/3 \cdot 2/3 \cdot 1/3 + 2/3 \cdot 1/6 \cdot 1/3}$$

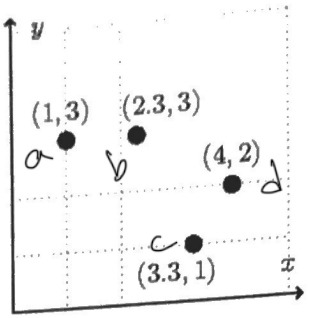
$$= \frac{2/9}{2/9 + 1/9} = \frac{2}{3}$$

Name: ID#:

Nearest Neighbor Regression, *(12 points.)*

For a regression problem to predict y given a scalar feature x, we observe training data (pictured at right):

	x	y
a	1	3
b	2.3	3
c	3.3	1
d	4	2



(1) Compute **training** MSE of a 1-nearest neighbor predictor. *(3 points.)*

ϕ

(2) Compute the **leave-one-out** cross-validation error (MSE) of a 1-nearest neighbor predictor. *(3 points.)*

	predict	error
a:	3	ϕ
b:	1	2
c:	2	1
d:	1	1

$MSE = \frac{1}{4} \cdot (2^2 + 1^2 + 1^2) = \frac{6}{4} = 1\frac{1}{2}$.

(3) Compute the **training** MSE of a 2-nearest neighbor predictor. *(3 points.)*

	predict	error
a:	3	ϕ
b:	2	1
c:	2.5	1.5
d:	1.5	.5

$MSE = \frac{1}{4}\left(1^2 + (\frac{3}{2})^2 + (\frac{1}{2})^2\right) = \frac{14}{16} = \frac{7}{8}$.

(4) Compute the **leave-one-out** cross-validation error (MSE) of a 2-nearest neighbor predictor. *(3 points.)*

	predict	error
a:	2	1
b:	2	1
c:	$2\frac{1}{2}$	$1\frac{1}{2}$
d:	2	ϕ

$MSE = \frac{1}{4}\left(1^2 + 1^2 + (\frac{3}{2})^2\right) = \frac{17}{16} = 1\frac{1}{16}$

Name: _____ ID#: _____

Gradient Descent, *(10 points.)*

Suppose that we have training data $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$, where $x^{(i)}$ is a scalar feature with which we wish to predict the real-valued target $y^{(i)}$. We decide to use a nonlinear regression model with two parameters, $\theta = [a, b]$:

$$\hat{y}(x) = f(x\,;\theta) = a + \exp(x_1 + b)$$

and train our model using gradient descent on the mean squared error (MSE) loss.

(1) Write down the gradient of our loss function. $MSE = \frac{1}{m}\sum (y-\hat{y})^2 = J(a,b)$

$$\nabla J = \left[\frac{\partial J}{\partial a} \quad \frac{\partial J}{\partial b}\right]$$

$$\frac{\partial J}{\partial a} = \frac{1}{m}\sum \left(y - (a + \exp(x_1+b))\right) \cdot 1$$

$$\frac{\partial J}{\partial b} = \frac{1}{m}\sum \left(y - (a + \exp(x_1+b))\right) \cdot (-1)\exp(x_1+b)$$

(2) Give one advantage of batch gradient descent over stochastic gradient

Many: Easier to set step size; monotonic; easier to gauge convergence.

(3) Give one advantage of stochastic gradient descent over batch gradient

Mostly: faster for large m (more steps per epoch)

May avoid small local optima

(4) Give pseudocode for a (batch) gradient descent function `theta = train(X,Y)`, including all necessary elements for it to work.

Eg.

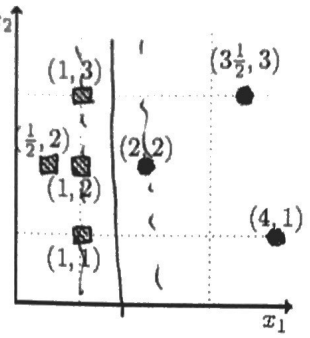
```
Init Θ = [a,b]
Set convergence tolerance ε.
do {
    Compute ∇J
    Select step size α
    Θ ← Θ − α ∇J
} while ( ||∇J|| > ε )
```

Name: _____ ID#: _____

Support Vector Machines, *(10 points.)*

Suppose we are learning a linear support vector machine with two real-valued features x_1, x_2 and binary target $y \in \{-1, +1\}$. We observe training data (pictured at right):

x_1	x_2	y
0.5	2	+1
1	1	+1
1	2	+1
1	3	+1
2	2	-1
3.5	3	-1
4	1	-1



Our linear classifier takes the form

$$f(x; w_1, w_2, b) = \text{sign}(w_1 x_1 + w_2 x_2 + b).$$

(1) Consider the optimal linear SVM classifier for the data, i.e., the one that separates the data and has the largest margin. **Sketch** its decision boundary in the above figure, and **list** the support vectors here. *(2 points.)* SVs are : (1,3) (1,2) (1,1) and (2,2)

(2) Derive the parameter values w_1, w_2, b of this $f(x)$ using these support vectors. What is the length of the margin? *(3 points.)*

Have $b + 1w_1 + 2w_2 = -1$ and $b + 2w_1 + 2w_2 = +1$

$b + 1w_1 + 1w_2 = -1$

$\Rightarrow w_1 = 2$

$\Rightarrow w_2 = 0$.

$\Rightarrow b = -3$

(3) What is the *training error* of a linear SVM on these data? *(2 points.)*

ϕ. (seperable)

(4) What is the the *leave-one-out cross validation error* for a linear SVM trained on these data? *(3 points.)*

1/7 - Decision boundary is the same if any point but (2,2) is removed.

Remove (2,2) $\Rightarrow$ boundary shifts to (at least) 2.25 $\Rightarrow$ error.

Name: _____ ID#: _____

VC-Dimensionality, *(10 points.)*

Consider the VC dimension of two classifiers defined using two features x_1, x_2.

(1) First, consider a simple classifier f_A that uses **only** x_1 and predicts class $+1$ "close to" a point μ:

$$f_A(x) = \begin{cases} +1 & \text{if } (x_1 - \mu_1)^2 < r \\ -1 & \text{otherwise} \end{cases}$$

What is the VC dimension of f_A? Justify your answer. *(5 points.)*

2 -

Decision function is a continuous interval from $\mu_1 - \sqrt{r}$ to $\mu_1 + \sqrt{r}$

Patterns + +  + −  etc

But, cannot shatter 3 : Pattern 

No contiguous interval w/ both +1's inside.

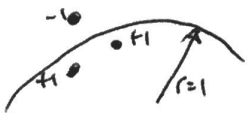
(2) Now, suppose that we use both x_1 and x_2, but keep the number of parameters of the model the same by forcing $r = 1$, giving the classifier:

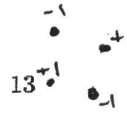
$$f_B(x) = \begin{cases} +1 & \text{if } ((x_1 - \mu_1)^2 + (x_2 - \mu_2)^2) < 1 \\ -1 & \text{otherwise} \end{cases}$$

What is the VC dimension of f_B? Justify your answer. *(5 points.)*

3 - Place points close together ($\|x^{(1)} - x^{(2)}\| << 1$).

Then  any one point is easy to separate from the others



But, 4 cannot :  similar to linear classifier.

CS273A Midterm Exam

Introduction to Machine Learning: Fall 2023

Monday November 6th, 2023

Your name: Peter Anteater

Row/Seat Number:

Your ID #(e.g., 123456789) 3141592653

UCINetID (e.g.ucinetid@uci.edu) panteat@uci.edu

- Please put your name and ID **on every page**.
- Total time is 80 minutes. 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 or TA to come over.
- You may use **one** sheet containing handwritten notes for reference, and a (basic) calculator.
- Turn in your notes and any scratch paper with your exam.

Problems

Total, *(75 points.)*

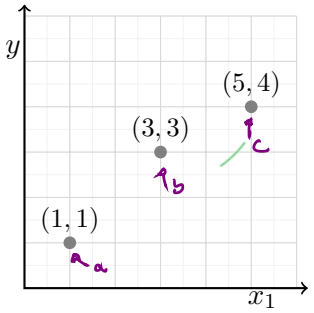
This page is intentionally blank, use as you wish.

Name: ID#:

Problem 1 Cross-Validation, *(15 points.)*

For a regression problem to predict real-valued y given a single real-valued feature x_1, we observe training data (pictured at right):

x_1	y
1.0	1.0
3.0	3.0
5.0	4.0



(1) Compute the **leave-one-out** cross-validation MSE of a 1-nearest neighbor predictor. (In case of ties, prefer to use the data listed earlier in the table.) *(3 points.)*

leave out	predict	err
a	3	2
b	1	2
c	3	1

$MSE = \frac{1}{3}(2^2 + 2^2 + 1^2)$

$= 3.$

(2) Compute the **leave-one-out** cross-validation MSE of a 2-nearest neighbor predictor. (In case of ties, prefer to use the data listed earlier in the table.) *(3 points.)*

out	pred	err
a	3.5	2.5
b	2.5	0.5
c	2	2

$MSE = \frac{1}{3}\left((5/2)^2 + (1/2)^2 + (4/2)^2\right)$

$= \frac{42}{12} = 3.5$

(3) Compute the **leave-one-out** cross-validation MSE of a constant predictor, $f(x) = \theta_0$. *(3 points.)*

out	pred	err
a	3.5	2.5
b	2.5	0.5
c	2	2

$MSE = 3.5$

(This happens to be the same as the 2-NN model, since the avg of the other 2 points is the same as the mean of all the other (2) points.)

(4) Compute the **leave-one-out** cross-validation MSE of a linear predictor, $f(x) = \theta_1 x_1 + \theta_0$. *(3 points.)*

out	pred	err
a	2	1
b	2.5	0.5
c	5	1

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

$= 9/12 = 3/4.$

(5) If choosing among these models, which would you select (based on cross-validation)? *(3 points.)*

We should choose the linear model (#4), since it has the smallest LOOXV MSE.

This page is intentionally blank, use as you wish.

Name: ID#:

Problem 2 True/False, *(10 points.)*

Here, assume that we have m data points $y^{(i)}$, $x^{(i)}$, $i = 1 \ldots m$, each with n features, $x^{(i)} = [x_1^{(i)} \ \ldots \ x_n^{(i)}]$. For each of the scenarios below, circle one of "true" or "false" to indicate whether you agree with the statement.

True or **false**: Using "batch" gradient descent (all m data per step) is less prone to getting stuck in local optima than stochastic gradient descent.

True or **false**: Optimizing a linear classifier using the hinge loss and L_2 regularization will always converge to a global optimum of the loss.

True or **false**: Increasing the number of features available to a perceptron model will increase its bias.

True or **false**: The perceptron algorithm is always guaranteed to converge.

True or **false**: With sufficiently many data, an SVM with polynomial kernel $K(x, x') = (1+x \cdot x'^T)^2$ can approximate any decision function.

True or **false**: The SVM optimization problem is an example of a linear program (a linear objective with linear constraints).

True or **false**: Using a soft-margin SVM in place of a hard-margin SVM will typically reduce the training error.

True or **false**: A 1-nearest neighbor model is an example of a learner with no inductive bias.

True or **false**: Feature selection (electing to ignore certain features of the data in our model) can be used to reduce model variance.

True or **false**: Given sufficiently many layers, a neural network with one hidden node per layer can approximate any function.

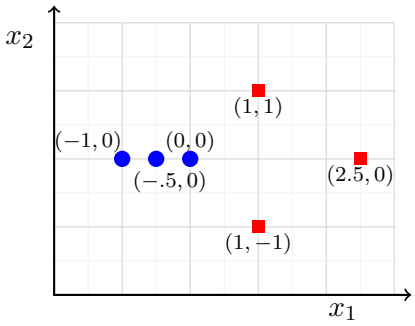
This page is intentionally blank, use as you wish.

Name: _____ ID#: _____

Problem 3 Support Vector Machines, *(12 points.)*

For a classification problem to predict binary y given two real-valued features x_1, x_2, we observe training data (pictured at right):

x_1	x_2	y
0.0	0.0	-1
-0.5	0.0	-1
-1.0	0.0	-1
1.0	-1.0	1
1.0	1.0	1
2.5	0.0	1



Our linear classifier takes the form,

$$f(x; w_1, w_2, b) = \text{sign}(w_1 x_1 + w_2 x_2 + b).$$

(1) Consider the optimal linear SVM classifier for the data, i.e., the one that separates the data and has the largest margin. **Sketch** its decision boundary in the above figure, and **list** the support vectors here. *(2 points.)*

SVs: (0,0), (1,1), (1,-1)

(2) Derive the parameter values w_1, w_2, b of this $f(x)$ using these support vectors. *(4 points.)*

$w_1 \cdot 0 + w_2 \cdot 0 + b = -1$

$w_1 \cdot 1 + w_2 \cdot 1 + b = +1$

$w_1 \cdot 1 + w_2 \cdot (-1) + b = +1$

$\Rightarrow$ $b = -1$, $w_2 = 0$, $w_1 = 2$

(3) What is the *training error rate* of a linear SVM on these data? *(3 points.)*

$\emptyset$

(4) What is the the *leave-one-out cross validation error rate* for a linear SVM trained on these data? *(3 points.)*

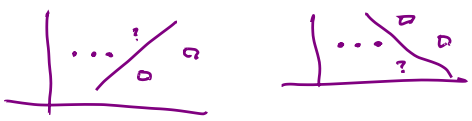
Non-SVs - all predict correctly ✓

(0,0) - ok: new boundary is at $x_1 = 1/2$

(1,1) - wrong;

(1,-1) - wrong:

$\Rightarrow$ Err = $2/6 = 1/3$



This page is intentionally blank, use as you wish.

Name: _____ ID#: _____

Problem 4 Naïve Bayes Classifiers, *(12 points.)*

x_1	x_2	y
b	0	0
c	1	0
a	0	0
a	1	0
c	0	1
c	1	1
c	1	1
a	0	1
c	0	1
c	1	1

Consider the table of measured data given at right. We will use the two observed features x_1 , x_2 to predict the class y. Feature x_1 can take on one of three values, $x_1 \in \{a, b, c\}$; feature x_2 is binary, $x_2 \in \{0, 1\}$. In the case of a tie, we will prefer to predict class $y = 0$.

(1) Write down the probabilities learned by a naïve Bayes classifier: *(4 points.)*

$p(y=0)$: $4/10$ $\quad$ $p(y=1)$: $6/10$

$p(x_1=a \mid y=0)$: $2/4$ $\quad$ $p(x_1=a \mid y=1)$: $1/6$

$p(x_1=b \mid y=0)$: $1/4$ $\quad$ $p(x_1=b \mid y=1)$: $\emptyset$

$p(x_1=c \mid y=0)$: $1/4$ $\quad$ $p(x_1=c \mid y=1)$: $5/6$

$p(x_2=0 \mid y=0)$: $1/2$ $\quad$ $p(x_2=0 \mid y=1)$: $1/2$

$p(x_2=1 \mid y=0)$: $1/2$ $\quad$ $p(x_2=1 \mid y=1)$: $1/2$

(2) Using your naïve Bayes model, compute the probability $p(y=1 \mid x_1=c, x_2=0)$: *(4 points.)*

$P_0 = p(y=0, x_1=c, x_2=0) = 4/10 \cdot 1/4 \cdot 1/2 = 1/20$

$P_1 = p(y=1, x_1=c, x_2=0) = 6/10 \cdot 5/6 \cdot 1/2 = 5/20$

$\Rightarrow p(y=1 \mid x_1=c, x_2=0) = \frac{5}{5+1} = 5/6$

(3) Using your naïve Bayes model, compute the probability $p(y=1 \mid x_1=b, x_2=1)$: *(4 points.)*

$P_0 = p(y=0, x_1=b, x_2=1) = 4/10 \cdot 1/4 \cdot 1/2 = 1/20$

$P_1 = p(y=1, x_1=b, x_2=1) = 6/10 \cdot \emptyset \cdot 1/2 = \emptyset$

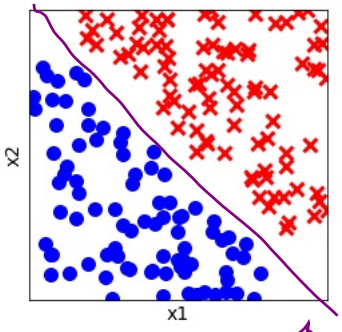
$\Rightarrow p(y=1 \mid x_1=b, x_2=1) = \emptyset.$

This page is intentionally blank, use as you wish.

Name: ID#:

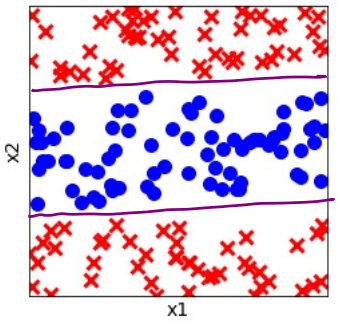
Problem 5 Separability, *(16 points.)*

Consider each of the following pictured data sets. Select **all** sets of features that could be used by a perceptron (linear classifier) to separate the data. (Note: check each feature list carefully; they may differ by subproblem.)



needs x_1 & x_2, anything else is extra.

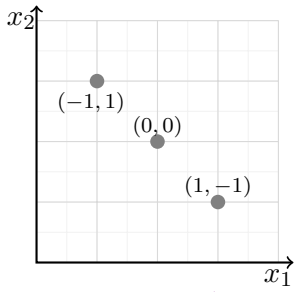
- ☐ $[1, x_1]$
- ☐ $[1, x_2]$
- ☑ $[1, x_1, x_2, (x_1)^2]$
- ☑ $[1, x_1, x_2, (x_2)^2]$
- ☑ $[1, x_1, x_2, (x_1)^2, (x_2)^2, (x_1 x_2)]$



needs decision boundary at two different values of x_2, but doesn't need to depend on x_1

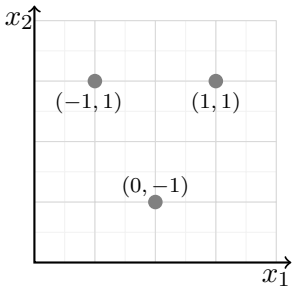
- ☐ $[1, x_1]$
- ☐ $[1, x_2]$
- ☐ $[1, x_1, x_2, (x_1)^2]$
- ☑ $[1, x_1, x_2, (x_2)^2]$
- ☑ $[1, x_1, x_2, (x_1)^2, (x_2)^2, (x_1 x_2)]$

Now consider the following sets of feature vectors, with no target labels. Select **all** the sets of features that could be used by a perceptron (linear classifier) to **shatter** the data, i.e., correctly separate any setting of the target values $y^{(i)}$:



collinear placement ⇒ can't shatter with $[1, x_1, x_2]$, but quad. function of either x_1 or x_2 can do it.

- ☐ $[1, x_1]$
- ☐ $[1, x_2]$
- ☐ $[1, x_1, x_2]$
- ☑ $[1, x_1, (x_1)^2]$
- ☑ $[1, x_2, (x_2)^2]$



placement ok for shattering with $[1, x_1, x_2]$, but two points with identical x_2 values means we can't do it without x_1.

- ☐ $[1, x_1]$
- ☐ $[1, x_2]$
- ☑ $[1, x_1, x_2]$
- ☑ $[1, x_1, (x_1)^2]$
- ☐ $[1, x_2, (x_2)^2]$

This page is intentionally blank, use as you wish.

Name: ID#:

Problem 6 Gradient Descent, *(10 points.)*

Suppose that we have training data $\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\}$, where $x^{(i)}$ is a scalar feature and $y^{(i)} \in \{-1, +1\}$, and we wish to train a linear classifier, $\hat{y} = \text{sign}[\, a + bx \,]$, with two parameters a, b. In order to train the model, we use stochastic gradient descent on the *exponential loss* surrogate, whose loss for data point i is given by:

$$J^{(i)}(X, Y) = \exp\left[\ -y^{(i)}(a + bx^{(i)})\ \right].$$

(1) Write down the gradient of the single-data surrogate loss $J^{(i)}$ *(4 points.)*:

$\frac{\partial J^i}{\partial a} = \exp\left[-y^i(a+bx^i)\right] \cdot (-y^i)$

$\frac{\partial J^i}{\partial b} = \exp\left[-y^i(a+bx^i)\right] \cdot (-y^i)(x^i)$

$\nabla J^i = \left[\frac{\partial J^i}{\partial a} \quad \frac{\partial J^i}{\partial b}\right]$.

(2) Give one advantage of stochastic gradient descent over batch gradient *(2 points.)*:

SGD is much faster at "early" optimization, since it performs many more parameter updates per epoch.

(3) Give pseudocode for a stochastic gradient descent function `theta = train(X,Y)`, including all necessary elements for it to work. *(4 points.)*

Init $\theta = \underline{0}$, or at random

Set step size $\alpha = .001$ (or something)

for epoch in 1... max:

  for each data point i, in some order (eg. random)

    $\theta \leftarrow \theta - \alpha \nabla J^i$

- We can use fancier stopping criteria, or alter the step size as we go, but this is enough to function.

This page is intentionally blank, use as you wish.

This page is intentionally blank, use as you wish.

This page is intentionally blank, use as you wish.

This page is intentionally blank, use as you wish.

This page is intentionally blank, use as you wish.