CMPS 242 Second Homework, Fall 2019

5 Problems. 14 pts, due 11:50pm Friday Oct. 18

This homework is to be done in groups of 2. Each group members should completely understand the group's solutions and *must* acknowledge all sources of inspiration, techniques, and/or helpful ideas (web, people, books, etc.) other than the instructor, TA, and class text. Each group should electronically submit a single set of solutions under **one** of the group member's names to canvas. The solution should have the names and e-mail addresses of both group members at the top. The other group member should submit only the name and E-mail address of the person submitting the actual solutions. Your solutions should be clearly numbered and in order. Although there are no explicit points for "neatness", the TA may deduct points for illegible or poorly organized solutions.

I *strongly* recommend that each student work on all of the problems individually before integrating their solutions into the group consensus. Dividing up the problems so each student does just a subset of them is the wrong approach.

1. (4 pts) Consider a learning problem where the domain is the 1000 consecutive integers $\{0, 1, 2, 3, \ldots, 999\}$, so each feature vector $\mathbf{x}$ is just an interger between 0 and 999. Let the hypothesis class be intervals (each hypothesis maps an interval of these points to + and everything else to −), so we assume the inductive bias that all the integers in some unknown interval are labeled "+", and all integers outside the interval are labeled "−". This problem asks you to run some simulations where the target concept maps the integers between 100 and 600 to + and everything else to −.

   (a) First (1 pt), assume the training data and test point are both drawn from the uniform distribution over the domain so each point has probability 1/1000. Consider the most-specific learning algorithm that always returns the smallest interval containing the positively labeled data in the training data as its hypothesis. This problem investigates the goodness of this learning algorithm's hypothesis as a function of the number of training examples.

   The error rate of a hypothesis is the probability that it mis-labels a new random instance. Assume that we can tolerate an error-rate of 0.05 (5% errors), so producing any hypothesis with a larger error-rate is a failure. **Use simulations** to estimate the probability that the most-specific algorithm has an error rate at most 0.05 (as a function of the training set) when trained on 10, 20, 30, etc. random examples by training on each sample size several times. Also record and report the average error rate you observe for the various training set sizes. How does the failure probability and average error rate change as a function of the number of random examples? Provide a table with your results.

   (b) Next (1 pt) assume the distribution is not uniform, but skewed towards the lower examples. To generate this distribution, sample five integers uniformly (and independently) from the [0..999] interval, and return the lowest integer. This will make low numbers more frequent and higher ones less so. Do you expect the

algorithm's hypotheses to be more or less accurate when both the training and test data are generated in this more complicated way? Repeat the simulation experiment above with this change in the distribution and report your results. (you can either evaluate the hypotheses's error rate analytically or use a large number of test random examples, say 50,000, to evaluate the errors).

(c) (1 pt) Evaluate the error rate of hypotheses generated using training sets drawn from one distribution but tested on the other distribution (again, empirically rather than analytically). How much does the error rate increase when the hypothesis is trained on the "wrong" distribution? Consider training set sizes in the range 50 to 100.

(d) Finally (1 pt), consider the hypotheses generated by the algorithm for the first part. **Assume** you wanted to find a (different) distribution for test data that made these hypotheses tend to perform very poorly on random test data from this different distribution. **Informally** describe what kinds of test distributions would tend to make these hypotheses perform very poorly. By picking the "worst" test distribution, roughly how large do you think you can make the likely test error rate when the hypothesis is generated by 100 random examples from the uniform distribution?

Optional (not to be turned in): What would change in the above problems if the learning algorithm used the most general hypothesis consistent with the sample? This hypothesis predicts "+" on the largest interval containing all the positive points, but none of the negative points.

2. (4 pts) This problem guides you through a very simple PAC-style (Probably Approximately Correct) generalization bound for a simple binary classification problem. This kind of bound says that with reasonably high probability, the learned hypothesis (prediction rule) will be reasonably accurate.

   Say your new significant other loves chocolate, but not chocolate that is too dark. Assume that the darkness of the chocolate (and indirectly your significant other's preferences) are accurately reflected by the percentage (from 0 to 100) of Cocoa solids in the package labeling. Although in real life, these precentages tend to be integers, we will assume that the cocoa percentages are real-valued. The goal is to predict if your significant other will like new chocolate bars you encounter.

   The batch learning framework postulates a distribution over candy bars, which in this case are measured simply by a scalar indicating their percentage of cocoa solids. We will use $p(x)$ for this (unknown) distribution over cocoa percentages. Assume that $p()$ is a continuous density function. We assume that the target function to be learned is encoded by some unknown threshold θ where the candy bar is tasty if the cocoa percentage is less than (or equal to) θ (call this label +), and not tasty (label "$-$") if greater than θ.

The learning problem in this case is to estimate a threshold $\hat{\theta}$ that approximates θ in the sense that its predictions are mostly accurate with respect to distribution $p(x)$, i.e. that $p((x \leq \hat{\theta}) \neq (x \leq \theta))$ is small, traditionally ϵ is used for this error probability. The learned estimate $\hat{\theta}$ is a function fo the training set, and some training sets may be uninformative. For example, if $p(x)$ is uniform on $(0, 100)$ even a large training set has some (perhaps very small) chance of only containing examples where $x \leq 20$, giving not much good information (assuming $\theta > 20$) on where θ actually lies. Therefore we allow the learning algorithm a (hopefully) small "failure" probability, traditionally measured by δ.

Similar to the previous problem, we will use consider the learning algorithm producing the most specific hypothesis, the one that sets $\hat{\theta}$ to the largest + point in the training data. **Call this learning algorithm** A. We call the produced hypothesis $h_{\hat{\theta}}$. Each example (x, t) is generated independently by first sampling from density $p()$ to get the feature value x and then setting the label t to be + if $x \leq \theta$ and $t = -$ if $x > \theta$. Thus there is no noise in the data, and $\hat{\theta} \leq \theta$.

We first notice that any $\hat{\theta}$ with $\hat{\theta} < \theta$ predicts incorrectly on a new random example (x, t) exactly when x falls in the half-open interval $(\hat{\theta}, \theta]$. Therefore the error rate of $h_{\hat{\theta}}$ is the probability under density p (which is unknown to the algorithm) of the interval $(\hat{\theta}, \theta]$.

Now consider an arbitrary error tolerance ϵ and let θ_ϵ be the value such that the interval $p((\theta_\epsilon, \theta]) = \epsilon$ (assume that $p((0, \theta]) > \epsilon$ so θ_ϵ exists).

(a) What is the probability that a randomly drawn point from $p()$ falls *outside* the interval $(\theta_\epsilon, \theta]$?

(b) What is the probability that *all* N of the random examples in **the training data** fall outside the interval $(\theta_\epsilon, \theta]$?

(c) What is the probability (over the randomly drawn N-example training set) that A produces an $h_{\hat{\theta}}$ with error rate at least ϵ?

(d) What is the smallest training set size N making the probability (with respect to the random training examples) that $h_{\hat{\theta}}$ has error greater than ϵ is at most δ? (i.e. the probability that $h_{\hat{\theta}}$ has error less than ϵ is at least $1 - \delta$, express N as a function of ϵ and δ).

3. (2 pts) Bayesian decision theory.

In some classification problems we have the option to *abstain* on an instance instead of predicting a particular class. For example, if we were to classify vehicles as either **B**usses, **C**ars, or **T**rucks, we might also include an **A**bstain prediction. The loss matrix would then become:

loss	prediction			
class	B	C	T	A
B	0	1	1	a
C	1	0	1	a
T	1	1	0	a

where a is the cost (or loss) of abstaining.

(a) Assume that on some instance x we know[1] $P(B \mid x) = 1/9$, $P(C \mid x) = 2/3$, and $P(T \mid x) = 2/9$. What is the *risk* (expected loss) of each of the predictions **B**, **C**, **T**, and **A**? Note that the risk of **A** will be a function of a.

What values of a makes abstaining have the lowest risk?

(b) Formulate a general prediction rule minimizing the risk that describes when to predict **B**, **C**, **T**, and **A** as a function of $P(B \mid x)$, $P(C \mid x)$ $P(T \mid x)$ and a.

What happens when a goes to 0?

4. (2 pts) Assume we have a two-class learning problem where the labeled data consists of 1,000 data points from one class, and 10 data points from the other class. A grad student is working on a machine learning algorithm and manages to get 99% accuracy using ten fold cross validation. Recall that 10-fold cross validation splits the data into 10 equal-sized chunks. For each chunk, the algorithm is trained on the other 9 and then tested on the held out chunk. The accuracies on each of the held-out chunks are then averaged to get an overall accuracy rate. Is this a big achievement for the algorithm? Justify your answer by describing why it should be difficult to get this accuracy on this kind of data, or giving a simple classification strategy whose accuracy is at least as good.

5. (2 pts) Gaussian generative models.

Assume that we are solving a two-class (0,1) classification problem using Gaussian generative models where the two distributions $P(\mathbf{x}|T=0)$ and $P(\mathbf{x}|T=1)$ share the same covariance matrix $\mathbf{\Sigma}$. Thus, on a new point $\mathbf{x}$ we predict label 1 if our learned $P(T=1)P(\mathbf{x}|T=1)$ is greater than $P(T=0)P(\mathbf{x}|T=0)$. Here $P(T)$ is estimated (learned) from the data, as is the shared covariance matrix $\mathbf{\Sigma}$ and the class means vectors $\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_0$. Furthermore,

$$P(\mathbf{x}|T=1) = \frac{1}{Z} \exp\left((\mathbf{x}-\boldsymbol{\mu}_1)^\top \mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_1)\right)$$

and similarly for $P(\mathbf{x}|T=0)$, both using the same $\mathbf{\Sigma}$ and normalization Z.

The *decision boundary* is the $\mathbf{x}$-values where the prediction changes, where $P(T=1)P(\mathbf{x}|T=1)$ equals $P(T=0)P(\mathbf{x}|T=0)$ in this case. Show (prove analytically) that

[1] In most situations we would only be able to estimate these probabilities, and thus would only be able to estimate the risks of different predictions.

the decision boundary is a hyperplane (i.e. described by $\mathbf{w}^\top \mathbf{x} = c$ for some vector $\mathbf{w}$ and constant c.

Hint: solve the equality for $\mathbf{x}$. Recall that if M is a $d \times d$ array and $\mathbf{u}$ and $\mathbf{v}$ are d-dimensional vectors then $\mathbf{u}^\top M \mathbf{v}$ is a scalar so:

$$\mathbf{u}^\top M \mathbf{v} = (\mathbf{u}^\top M \mathbf{v})^\top = \mathbf{v}^\top M^\top \mathbf{u}.$$

Furthermore, covariance matrices and their inverses are symmetric, so $(\mathbf{\Sigma}^{-1})^\top = \mathbf{\Sigma}^{-1}$.