CSC 580 Homework 1

Due: 9/14 (W) 5pm

Instructions:

- Submit your homework on time to gradescope. NO LATE DAYS, NO LATE SUBMISSIONS ACCEPTED.
- The submission must be one single PDF file (use Acrobat Pro from the UA software library if you need to merge multiple PDFs).
- Email your code to csc580homeworks@gmail.com.
  - You can use word processing software like Microsoft Word or LaTeX.
  - You can also hand-write your answers and then scan it. If you use your phone camera, I recommend using TurboScan (smartphone app) or similar ones to avoid looking slanted or showing the background.
  - Watch the video and follow the instruction: https://youtu.be/KMPoby5g_nE .
  - Points will deducted when you do not follow the instruction.
- Collaboration policy: do not discuss answers with your classmates. You can discuss HW for the clarification or any math/programming issues at a high-level. If you do get help from someone, please make sure you write their names down in your answer.
- If you cannot answer a problem, describing what efforts you have put in to solve the problem and where you get stuck will receive partial credit. Also, feel free to post your questions on Piazza.

Problem 1. Let (X, Y) follow the distribution $\mathcal{D}$. In this problem, you can assume that X, Y take values from discrete sets $\mathcal{X}$ and $\{1, \ldots, C\}$, respectively.

(a) Fix an instance x. What is the error rate of the prediction rule $f(x) = \arg\max_{y \in \mathcal{Y}} \mathbb{P}(Y = y \mid X = x)$ on the instance x, i.e., $\mathbb{P}(f(X) \neq Y \mid X = x)$? Show your derivation.

(b) Now, suppose you know $\mathcal{D}$. Given an feature vector x, suppose that your prediction is $\hat{Y} \in \{1, \ldots, C\}$ where $\hat{Y}$ is an independent sample from a distribution whose probability mass function is $\mathbb{P}(Y = y \mid X = x), y \in \{1, \ldots, C\}$. What is the error rate of your prediction on the instance x?

(c) Let $f^{(a)}$ be the prediction rule described by (a) above. Define $f^{(b)}$ similarly. Define $\mathsf{err}(f) = \mathbb{E}_{(X,Y)\sim\mathcal{D}}[\mathbb{1}\{f(X) \neq Y\}]$. Compute $\mathsf{err}(f^{(a)})$ and $\mathsf{err}(f^{(b)})$. Is $\mathsf{err}(f^{(a)}) \leq \mathsf{err}(f^{(b)})$? Why?

Problem 2. Generalized uncertainty measures. Throughout, assume the label space is $\mathcal{Y} = \{1, 2, 3\}$ and the feature x_f's are binary.

(a) For $\mathbf{p} = (p_1, p_2, p_3)$, recall that the classification error-based uncertainty measure is defined as $v_1(\mathbf{p}) = 1 - \max_{k=1}^{3} p_k$, and the Gini index-based uncertainty measure is defined as $v_2(\mathbf{p}) = 1 - \sum_{k=1}^{d} p_k^2$.

Let $\mathbf{p} = (0.6, 0, 0.4)$ and $\mathbf{q} = (0.62, 0.19, 0.19)$. Is $u_1(\mathbf{p}) \geq u_1(\mathbf{q})$? Is $u_2(\mathbf{p}) \geq u_2(\mathbf{q})$? Why?

(b) Given a general uncertainty function u and a labeled dataset S, if feature x_f is independent of y under P_S, then what is the value of the uncertainty reduction score of feature f, $\mathrm{Score}(f, S)$?

(c) Intuitively, we would like the uncertainty reduction scores to be generally nonnegative. Luckily we can construct uncertainty function with such properties using concave functions:

Definition 1. A function v is said to be *concave*, if for any probability vectors $\mathbf{p}, \mathbf{q}$ and any $t \in [0, 1]$, $v(t\mathbf{p} + (1-t)\mathbf{q}) \geq tv(\mathbf{p}) + (1-t)v(\mathbf{q})$.

It can be checked that v_1, v_2 above, as well as the entropy function, are all concave (this is beyond the scope of this exercise). Prove that: if the uncertainty function is of the form $u(S) = v(P_S(y = 1), P_S(y = 2), P_S(y = 3))$ for some concave function v, then for any feature f, $\mathrm{Score}(f, S) \geq 0$.

Problem 3. Decision trees with entropy uncertainty.

(a) Consider the entropy uncertainty $u(S) = \sum_{y \in \mathcal{Y}} P_S(Y = y) \log_2(\frac{1}{P_S(Y=y)})$ where S is a labeled dataset and $P_S(Y = y)$ is the fraction of examples in S with label y. Note that when $P_S(Y = y) = 0$, the term $V_y = P_S(Y = y) \log_2(1/P_S(Y = y))$ is undefined. In this case, which value should we use for V_y to make sure V_y is continuous w.r.t. $P_S(Y = y)$?

(b) Implement the decision tree in python as described in the book (handles only the binary features) but use the entropy instead of the classification error. Implement an option of `max_depth` so the trained tree will have depth at most `max_depth` (in our case, it corresponds to only considering at most `max_depth` features).

Use the data in the book (Table 1) while taking the rating 2/1/0 as positive and -1/-2 as negative. Train your decision tree with your code with `max_depth` = 2. Report your tree along with the following information (in whatever form a person can reasonably comprehend)

- What are the branching questions at each node?
- What are the uncertainty scores at each node?
- Show the predicted label for each leaf node.

Problem 4. The Bayes optimal classifier.

Let us define the data distribution $\mathcal{D}$ for the feature space $\mathcal{X}$ and the label space $\mathcal{Y}$ as follows. Consider the data distribution where the features $X \in \mathbb{R}^2$ is drawn so that $X_1 \sim \mathsf{Uniform}[0,1]$ and $X_2 \sim \mathsf{Uniform}[0,1]$. For $x \in \mathbb{R}^2$, let $i(x)$ be 1 if $x(1) < 1/3$, 2 if $x(1) \in [1/3, 2/3)$, and 3 if $x(1) \geq 2/3$. Define $j(x)$ similarly for $x(2)$ (i.e., replace $x(1)$ above by $x(2)$). Furthermore, the labels are either 0 or 1 where $\mathbb{P}(Y = 1 \mid X = x) = A_{i(x),j(x)}$ where

$$A = \begin{pmatrix} .1 & .2 & .2 \\ .2 & .4 & .8 \\ .2 & .8 & .9 \end{pmatrix}$$

($A_{i,j} \in \mathbb{R}$ is the entry of matrix A at i-th row, j-th column) Throughout, we abuse notation and use $\mathbb{P}$ for both probability of events and the density function for continuous random variables.

(a) What is the Bayes optimal classifier of $\mathcal{D}$ (describe its decision rule with English+math)? What is the error rate of the Bayes optimal classifier?

(b) Suppose we have access to x_1 only (and not to x_2). This induces a marginal data distribution over (X_1, Y); denote it by $\mathcal{D}_1$. Describe $\mathcal{D}_1$; specifically, provide $\mathbb{P}_{\mathcal{D}_1}(X_1 = x(1))$ and $\mathbb{P}_{\mathcal{D}_1}(Y = y \mid X_1 = x(1))$.

(c) Answer the same questions as (a) but now with $\mathcal{D}_1$.

(d) Draw the decision surface for the Bayes optimal classifier of $\mathcal{D}$. See https://scikit-learn.org/stable/auto_examples/tree/plot_iris_dtc.html for an example.

Problem 5. Hyperparameter tuning for k-NN.

In this problem, you can use existing k-NN implementations (e.g., scikit-learn), but you will get +10pts if you implement it by yourself; make sure you explicitly mention your choice in your solutions.

Consider the data distribution $\mathcal{D}$ from Problem 4. Draw 10,000 points from $\mathcal{D}$ and call them a test set, but do this once and for all and use the same test set throughout the problems here.

(a) Implement a function that draws m i.i.d. samples from $\mathcal{D}$.

(b) Let us plot a so-called *learning curve* for nearest-neighbor classification. Let $k = 4$. Define $\mathcal{M} = \{10, 30, 100, 300, 1000, 3000\}$. Call the following one 'trial':

- Draw 3000 fresh data points from $\mathcal{D}$ and call it S.
- Then, for each $m \in \mathcal{M}$, choose the first m data points from S, train a k-NN classifier with them, and then evaluate its test set error.

Perform 5 trials, compute the average test set error, and report the plot of 'test error rate' vs m. In the same plot, plot a horizontal line that shows the Bayes error rate so we know how close we get to the Bayes error rate.

(c) Let $\mathcal{K} = \{1, 2, 4, 8, 16, 32, 64\}$. Do (b) for every $k \in \mathcal{K}$. When $k \geq m$, simply force the code to set $k = m$.

(d) Let us add hyperparameter tuning to the procedure in (b). For this, just perform one trial (instead of 5) for simplicity. For each $m \in \mathcal{M}$, try tuning k by each of the following:

- training set error.
- 20% hold out from the training set.
- 5-fold cross validation.
- test set error (this is impossible in practice, but we just want to see what is the actually best one).

Report: for each $m \in \mathcal{M}$ and each tuning method,

- What is the tuned k?
- What is the test set error when you use the tuned k? How far are they from the actual best k measured by the test set tuning?

Discuss your findings; e.g., does one performs better than the other? why? if there is a failing method, explain why.

(e) Let $\mathcal{A}$ be a learning algorithm that takes in a dataset S and performs 5-fold cross validation with k-NN to choose the best $k \in \{1, 2, 4, 8, 16, 32, 64\}$, and then trains a k-NN classifier using S with that chosen k. Plot the learning curve of $\mathcal{A}$. For each $m \in \mathcal{M}$, plot both the training set data points and the decision surface of the classifier obtained by $\mathcal{A}$ in one figure. (Total $|\mathcal{M}|$ plots.)

(f) Use your decision tree learning algorithm written in Problem 3, and perform the same procedure as (e), but now by tuning its `max_depth` $\in \{0, 1, 2, 3, 4, 5, 6\}$. Compare the answers from (e) and (f) side by side. Discuss how they are different and if so what might be the cause of the difference. Are there any qualitative difference in the decision boundaries?