COMPSCI 671D Fall 2019
Homework 1

1 Perceptron (25 points)

Consider applying the perceptron algorithm (through the origin) on a dataset $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$. The algorithm starts with one data point and then cycles through the data until it makes no further mistakes. The algorithm starts with $\mathbf{w} = \mathbf{0}$. Due to the ambiguity of which side of the decision boundary a point is on when it is on the decision boundary itself, when $\mathbf{w}^T\mathbf{x}_i = \mathbf{0}$, the algorithm will regard this as making a mistake on point i.

(a) The dataset is $\mathbf{x}_1 = [1, 1]$, $\mathbf{x}_2 = [0, -1]$, and $\mathbf{x}_3 = [-2, 1]$ with labels $y_1 = 1$, $y_2 = -1$, and $y_3 = 1$. How many mistakes will the algorithm make when it starts cycling through the points, starting from $\mathbf{x}_1$? What about if it starts from $\mathbf{x}_2$ instead?

(b) Now assume that $\mathbf{x}_1$ is $[1, 0]$ instead of $[1, 1]$ while $\mathbf{x}_2$, $\mathbf{x}_3$ remain unchanged from **(a)**. How many mistakes will the algorithm make when it starts from $\mathbf{x}_1$? What if $\mathbf{x}_3 = [-10, 1]$ and $\mathbf{x}_1$, $\mathbf{x}_2$ remain unchanged from **(a)** (again starting from $\mathbf{x}_1$ and $\mathbf{x}_1$ is $[1, 1]$)?

(c) You may observe that the number of mistakes made in part **(b)** is larger than **(a)**. In lecture, we derive a convergence bound that the number of mistakes T made by the perceptron algorithm is upper-bounded by

$$T \leq \frac{1}{\delta^2}$$

where for all i, $y_i \mathbf{w}^T \mathbf{x}_i \geq \delta$ and $\max_i \|\mathbf{x}_i\|_2 = 1$. Although this is just an upper bound, it can still reflect the actual number of total mistakes made by the algorithm. How do the two changes (which are that $\mathbf{x}_1$ becomes $[1, 0]$ and $\mathbf{x}_3$ becomes $[-10, 1]$) of the datasets affect δ respectively? You need to calculate δ in both cases and thus you'll need to understand what δ actually means – plotting these points might be helpful. Now can you explain why these changes might increase/decrease the total mistakes made by the algorithm?

(d) Suppose our algorithm can select the data points in any order we want. Briefly describe an adversarial procedure for selecting the order of labeled data points so as to approximately maximize the number of mistakes the perceptron algorithm makes before converging. Assume that the data is indeed linearly separable, by a hyperplane that you (but not the algorithm) know. (You might get some inspiration from **(c)**)

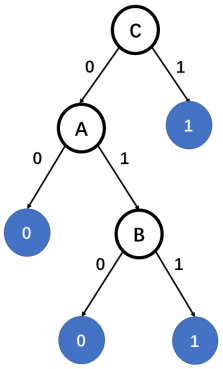

Figure 1: Decision tree representing $(A \wedge B) \vee C$

2 Decision Trees (25 points)

One great thing about decision trees is that they can represent logical functions. For example, the logical rule $(A \wedge B) \vee C$ can be represented as the decision tree in Figure 1.

(a) Please write down a decision tree representing the following logical rule

$$(A \wedge B) \vee (\bar{A} \wedge C)$$

Please make sure that your decision tree has the least possible number of nodes.

(b) As we learned from class, decision trees can be learned from data. The following table shows a dataset that forms the logical rule $(A \wedge B) \vee (\bar{A} \wedge C)$.

A	B	C	$(A \wedge B) \vee (\bar{A} \wedge C)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Please create a C4.5 decision tree based on this dataset (C4.5 uses Information Gain as the splitting criteria). If two features have the same Information Gain, please the choose the first one in lexographic order (choose A over B).

(c) Is the decision tree in **(b)** the same as the tree in **(a)**? What does this tell you about the C4.5 algorithm?

3 Evaluation Metrics for Imbalanced Data (30 points)

As stated by Chawla, Japkowicz, and Kolcz, "The class imbalance problem is pervasive and ubiquitous, causing trouble to a large segment of the data mining community." They are right: class

imbalance is sometimes very hard to deal with. One cannot usually optimize accuracy when dealing with an imbalanced dataset, because a dataset with only 1% positives will yield 99% accuracy if we simply classify all points as negative. In dealing with imbalanced data, different metrics rather than accuracy are often used.

(a) A simple way of dealing with imbalanced data is resampling the data such that the dataset is more balanced. Suppose the number of positives in the dataset is P and the number of negatives is N. This simple approach creates new points by drawing from the positive samples with probability $\frac{1}{2}$ and from the negative samples with probability $\frac{1}{2}$ and samples within class are evenly drawn.

Prove that for evaluation, the expected accuracy of a classifier tested on the resampled dataset is the same as its balanced accuracy tested on the original dataset. The balanced accuracy is defined as

$$Acc_{balance} := \frac{TPR + TNR}{2}.$$

(b) In lecture, AUC is defined as the area under ROC. An important property of AUC is that it equals the probability that a positive sample has larger predicted score than a negative one, where probability is calculated with respect to the draw of positive-negative pairs from the dataset. Please prove this theorem. For simplicity, just assume that the predicted score of all samples are different in order to avoid ties.

(c) The balanced accuracy is better than pure accuracy in dealing with imbalanced data, because it pays more attention to the minority class and less attention to the majority. Does the AUC have the same effect on imbalanced data? Why? You may want to use the theorem proved in **(b)** to explain it.

(d) Illustrate that AUC and balanced accuracy are different by giving an example when balanced accuracy disagrees with AUC, that is AUC of model A is greater than model B, but their balanced accuracies are in reverse order.

4 Validation and Testing (20 points)

For this problem we will provide you with a dataset and some sample code.

(a) Please use k-fold nested cross validation to select a hyperparameter for the model. In the sample code, a logistic regression model is trained on the UCI-Blood Transfusion Service Center Dataset. You may not know what logistic regression is yet (we'll get to that later in the semester), but you can just treat it as a black box that can be trained on datasets and make predictions. The sample code is in Python and uses scikit-learn, a commonly used Machine Learning toolkit. Please implement a 5-fold nested cross validation and choose the regularization parameter C from $\{0.1, 1, 10, 100\}$. The cross validation should be implemented by yourself rather than using scikit-learn or other toolkits. You should use F1 score for evaluation.

One thing you will find out in practice is that nested CV is very very computationally expensive, so you will have to make some kind of shortcut unless you want to wait for a long time for larger datasets. In general we will accept answers that use intelligent shortcuts. Possible shortcuts might involve determining earlier that certain parameter values don't work very well and omitting them for instance, or using subsamples of data to omit parameters before running on the full dataset. Luckily this dataset is very small, so you can actually do full nested CV.

(b) Suppose you need to choose the hyperparameter from two possible values C_1 and C_2. From experiments you find that C_1 outperforms C_2 in validation but C_2 performs better on the test set. Which one should you use in practice? Why? In answering this question, you may want to explain the difference between validation and testing.

(c) In **(a)** you may notice that the scale of features vary a lot in the Blood Transfusion Dataset. To achieve better performance, people usually normalize the features such that each column will have mean 0 and standard deviation 1. This is the same as the following: for each column, subtract its mean and dividing by its standard deviation. Write down the formulas explicitly for training normalization, and also write down the formulas for any normalization you wish to perform during validation and testing, if any.