CS 422/622 Final Exam

Name ________________________________

Instructions

This final exam consists of 13 questions: 7 5-point questions, 5 10-point questions and 1 15-point question. The exam is "choose your own adventure" – meaning that you must only complete 80 points worth of questions. The exam will be graded out of 80 and the questions that result in your best 80 points will be used. You have several options:

1. Skip the 15pt question and one 5pt question
2. Skip two 10pt questions
3. Skip four 5pt questions
4. Skip two 5pt questions and one 10pt question

STATE YOUR ASSUMPTIONS!

Short Answer (5 Points Each)

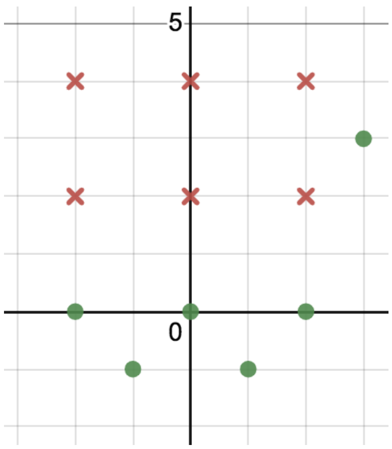
1. My dataset consists of 10 million samples, each with 10 real-valued features. Assume the data is labeled. What is the best algorithm/model for classifying this data? What is the worst? Briefly explain.

2. True/False: The VC Dimension of KNN with K=1 is infinite. Explain.

Short Answer (5 Points Each)

3. Which of the following models can achieve 100% accuracy on the training data below? Circle your choices. No need for explanation (unless you're unsure and want partial credit).

   - 1-NN
   - 3-NN
   - Decision Tree
   - SVM with Kernels
   - Soft-Margin SVM
   - Perceptron



4. Briefly explain, in English, each term in the following SVM formulation.

$$\min_{w,b} \frac{1}{\gamma(w,b)} + C\sum_n \xi_n$$

$$\text{subject to } y_n(w \cdot x_n + b) \geq 1 - \xi_n$$

$$\xi_n \geq 0$$

Short Answer (5 Points Each)

5. I have a dataset with 10,000 features. This is computationally expensive to deal with at train and test time. How can I use a supervised learning approach to identify only the most important features for training and testing?

6. Explain how the traditional lecture format in school is just an imitation learning problem and how it could be adjusted to better reflect a reinforcement learning environment.

Short Answer (5 Points)

7. Given the following two real-world examples, should recall and precision be weighted equally? Or should one be weighted more over the other? Briefly explain for each. This whole problem (a and b) counts for 5 points total.

   (a) Breast cancer detection from mammogram images.

   (b) Cybersecurity threat detection that results in an immediate system-wide shutdown.

Long Answer (10 Points)

8. Rank the following models from lowest to highest memory/computation cost at train and test time (lowest at the top, highest at the bottom): Decision Trees, K-NN, K-Means, Deep Neural Networks, and Perceptron. Make sure to state any assumptions, and explain your choices.

Train Memory	Train Computation	Test Memory	Test Computation

Long Answer (10 Points)

9. Give an example of a set of binary-labeled data for which each of the following classifiers would produce the exact same decision boundary: 1) Hard-Margin SVM, 2) Decision Tree, 3) Perceptron, 4) KNN with $K = 1$, and 5) K-Means with $K = 2$. Explain.

Long Answer (10 Points)

10. Can you exactly replicate a fully connected NN using a CNN? If not, explain. If yes, give an example to demonstrate.

Long Answer (10 Points)

11. For each of the following algorithms/models, give a brief explanation of how each handles outliers and duplicates:

    - Decision Trees
    - K-NN
    - Perceptron
    - SVM
    - Neural Networks

Long Answer (10 Points)

12. Deep learning is a rapidly growing technique in machine learning that makes very good predictions, but is not really able to explain why it made any particular prediction. If it's true that humans are likely unaware of their true motives for acting, should we demand machines be better at this than we actually are? Explain your reasoning.

Long Answer (15 Points)

13. How would you adjust the the following algorithms for regression? How about multi-class classification? The algorithms are provided on the back of the exam. Be specific.

    - K-NN

    - Perceptron

    - Decision Trees

Equations & Algorithms

Algorithm 1 DecisionTreeTrain(*data*, *remaining features*)

```
guess ← most frequent answer in data                      // default answer for this data
if the labels in data are unambiguous then
    return Leaf(guess)                                    // base case: no need to split further
else if remaining features is empty then
    return Leaf(guess)                                    // base case: cannot split further
else                                                      // we need to query more features
    for all f ∈ remaining features do
        NO ← the subset of data on which f=no
        YES ← the subset of data on which f=yes
        score[f] ← # of majority vote answers in NO
                  + # of majority vote answers in YES
                              // the accuracy we would get if we only queried on f
    end for
    f ← the feature with maximal score(f)
    NO ← the subset of data on which f=no
    YES ← the subset of data on which f=yes
    left ← DecisionTreeTrain(NO, remaining features \ {f})
    right ← DecisionTreeTrain(YES, remaining features \ {f})
    return Node(f, left, right)
end if
```

Algorithm 2 DecisionTreeTest(*tree*, *test point*)

```
if tree is of the form Leaf(guess) then
    return guess
else if tree is of the form Node(f, left, right) then
    if f = no in test point then
        return DecisionTreeTest(left, test point)
    else
        return DecisionTreeTest(right, test point)
    end if
end if
```

Algorithm 5 PerceptronTrain($\mathbf{D}$, *MaxIter*)

```
w_d ← 0, for all d = 1 … D                                // initialize weights
b ← 0                                                     // initialize bias
for iter = 1 … MaxIter do
    for all (x,y) ∈ D do
        a ← Σ_{d=1}^{D} w_d x_d + b                        // compute activation for this example
        if ya ≤ 0 then
            w_d ← w_d + y x_d, for all d = 1 … D           // update weights
            b ← b + y                                     // update bias
        end if
    end for
end for
return w_0, w_1, …, w_D, b
```

Algorithm 6 PerceptronTest($w_0, w_1, \dots, w_D, b, \hat{\mathbf{x}}$)

```
a ← Σ_{d=1}^{D} w_d x̂_d + b                                // compute activation for the test example
return sign(a)
```

Algorithm 3 KNN-Predict($\mathbf{D}$, K, $\hat{\mathbf{x}}$)

```
S ← [ ]
for n = 1 to N do
    S ← S ⊕ ⟨d(x_n, x̂), n⟩                                // store distance to training example n
end for
S ← sort(S)                                               // put lowest-distance objects first
ŷ ← 0
for k = 1 to K do
    ⟨dist,n⟩ ← S_k                                        // n this is the kth closest data point
    ŷ ← ŷ + y_n                               // vote according to the label for the nth training point
end for
return sign(ŷ)                                            // return +1 if ŷ > 0 and −1 if ŷ < 0
```