CSE 142 – Machine Learning, Fall 2021

Assignment #2
Due Friday, Nov 5th by 11:59pm PT

Notes:

- *This assignment is to be done individually. You may discuss the problems at a general level with others in the class (e.g., about the concepts underlying the question, or what lecture or reading material may be relevant), but the work you turn in must be solely your own.*
- Be sure to re-read the "Policy on Academic Integrity" on the course syllabus.
- Be aware of the late policy in the course syllabus.
- Justify every answer you give – **show the work** that achieves the answer or **explain** your response.
- Any updates or corrections will be posted both on Piazza and Canvas, so check there occasionally.
- To turn in your assignment:
  - Submission through Gradescope.
  - Clearly indicate which part of your submission corresponds to which question when submitting. If you don't do it right, grader might see "no content is available".
  - Submit your code for Problem #6 on Codalab. See the problem description for details.

Problem #1 [10 points]
We'd like to learn a Boolean function that separates the people in the Hatfield family from people in the McCoy family. We know the following information about a given person:

Age status: { *Child, Adult, Elderly* }
Residency: { *West Virginia, Kentucky* }
Teeth: { *Few, Many* }
Sympathizes with: { *Union, Confederate, Neither* }
Occupation: { *Miner, Bootlegger, Other, None* }

(a) For this simple problem, if testing a classification hypothesis takes a nanosecond, how long would it take to test every possible hypothesis?
(b) How long would it take if we used a conjunctive hypothesis space representation?
(c) How long would it take if we used a conjunctive hypothesis space with internal disjunctions?

Problem #2 [16 points]

The linear discriminant function for a binary classification problem with three features is the plane defined by the equation $3x_1 + 2x_2 + 4x_3 = 18$, where the features are x_1, x_2, and x_3. The discriminant plane separates the positive hypotheses ($3x_1 + 2x_2 + 4x_3 \geq 18$) from the negative hypotheses ($3x_1 + 2x_2 + 4x_3 < 18$).

The training set consists of the following labeled examples:

- (2, 2, 3) + (positive class)
- (3, 3, 2) +
- (1, 2, 3) +
- (1, 4, 1) +
- (4, 4, 4) +
- (2, 2, 2) +
- (3, 3, 1) – (negative class)
- (1, 1, 1) –
- (3, 3, 2) –
- (0, 4, 2) –
- (4, 0, 0) –
- (0, 0, 3) –

For each data point, give (a) its margin, (b) its 0-1 loss, (c) its hinge loss, and (d) its squared loss (clamped to zero above 1).

Problem #3 [10 points]

In a 5-class classification problem, 30 training examples are supplied that have the following class labels:

3	1	2	2	3	5
5	5	3	3	5	3
5	3	5	3	5	2
2	5	5	5	2	1
3	5	1	3	5	2

From the training data, we wish to estimate the probabilities of each class. Empirically estimate each class probability using (a) relative frequency, (b) Laplace correction, (c) m-estimate with m=5 and an even distribution of the pseudocounts, and (d) m-estimate with m=20 and an even distribution of the pseudocounts.

Plot these empirical probabilities, with class number on the x-axis and estimated probability on the y-axis – i.e., four plots (one for each approach), each of which consists of five connected points (the estimated class probabilities). Plot all four distributions on a single graph and label them.

Describe the trend as we go from (a) to (b) to (c) to (d) – that is, what does increasing the number of pseudocounts do (in general) to the probability distribution?

Problem #4 and **#5** use the following features to learn the concept *GoodMovie*, which predicts whether a movie is good or not based on several attributes:

Movie's budget = { Low, Medium, High }
Genre = { Documentary, Drama, Comedy }
Famous actors = { No, Yes }
Director = { Unknown, Great }

The training data for *GoodMovie* is:

#	*Budget*	*Genre*	*FamousActors*	*Director*	*GoodMovie*
1	Low	Documentary	Yes	Unknown	No
2	Low	Documentary	Yes	Great	No
3	Medium	Documentary	Yes	Unknown	Yes
4	High	Drama	Yes	Unknown	Yes
5	High	Comedy	No	Unknown	No
6	High	Comedy	No	Great	Yes
7	Medium	Comedy	No	Great	No
8	Medium	Documentary	No	Unknown	No
9	Low	Comedy	Yes	Great	Yes
10	Low	Drama	Yes	Unknown	No
11	Low	Comedy	No	Unknown	No
12	High	Drama	No	Unknown	Yes
13	Low	Drama	No	Great	Yes
14	Medium	Drama	Yes	Great	Yes
15	Medium	Documentary	No	Unknown	Yes
16	High	Drama	Yes	Great	Yes

Problem #4 [12 points]
In a conjunctive hypothesis space learning approach applied to the *GoodMovie* scenario above (basic CHS, not including disjunctions):

(a) How many possible (conjunctive) hypotheses are there?
(b) What is the least general hypothesis for *GoodMovie* after observing training data points 13 and 16 (only)?
(c) What is the most general hypothesis for *GoodMovie* after observing training data points 13, 16, and 7 (only)?
(d) What is the least general hypothesis for *GoodMovie* after observing training data points 10, 11, and 13 (only)?

Problem #5 [20 points]
Based on the GrowTree and BestSplit-Class algorithms and using the entropy impurity function, create a decision tree to learn the *GoodMovie* concept. Show how each node is decided (based on comparing impurity measures), then draw the full decision tree.

If there are ties in impurity measures, give higher priority to attributes and values according to their order on page 1: that is, Budget is the highest priority feature and Director is the lowest priority; within Budget, Low is the highest priority value, followed by Medium and then High. [Normally these might be randomly chosen, but we'll use this "inductive bias."] If there are ties to determine a leaf's label, give priority to "Yes."

Now apply this learned concept (decision tree) to the three test data sets posted on the Assignments page, and list the correctly and incorrectly classified examples (by number), for each test data set.

What is the error rate of the decision tree on the training data? On each test data set?

Problem #6 [40 points]
You need to sign up for CodaLab to submit problem #6. The email address of your CodaLab account should be your UCSC email, and the username will be your CruzID. ***The secret url*** *for CodaLab competition is* *https://competitions.codalab.org/competitions/35876?secret_key=deb3fa0a-039f-4b6f-a636-ea18b11591c8, where you can submit your code and get evaluation scores that will be listed on a leader-board. If you are not familiar with CodaLab competitions, check out it here.*
Note: ***for and only for*** *late submissions:* *https://competitions.codalab.org/competitions/35878?secret_key=12a6f3e0-1bbe-4f9d-ab22-683b408b385a*

In this problem, you will need to use the given training data to train a three-class linear classifier. Write a Python program named **`run.py`** that implements a **run(train_input_dir, train_label_dir, test_input_dir, pred_file)** function:
Parameters:

- *train_input_dir*: str - the path to the training dataset txt file.
- *train_label_dir* - the path to training dataset label txt file.
- *test_input_dir*: str - the path to the test dataset txt file.
- *pred_file*: str - the file name of your output prediction file.

The function should execute the following steps:

- 1. Load the training data examples located at train_dir.
- 2. Train a three-class linear classifier on the training data.
- 3. Load the test data from `test_dir` and make predictions using the classifier you trained. The prediction should be a numpy array with shape (N, 1).
- 4. Write the predictions into a single file named `pred_file`. Hint: you may use the function numpy.savetxt(pred_file, prediction, fmt='%1d', delimiter=",") where pred_file is the file name of your prediction, prediction is the numpy array of your prediction.

You don't need to know what the values of *train_input_dir*, *train_label_dir*, *test_input_dir* and *pred_file* will be. We will automatically import these variables on the Codalab platform.

Data Format:

- Training Data: two txt files, one with (N_0 + N_1 + N_2) per line (training) and one with class label (training_label).
    - In dataset 1 and 2 the input will have three input features in *random order*, while hidden dataset 3 will have four input features.
    - Please see the training1.txt, training1_label.txt for an example.

- Test Data: two txt files with (N_0 + N_1 + N_2) per line (testing) and one with class label (test_label).
    - The format of test data is the same as training data, except that all the feature vectors will be in *random order*.
    - When evaluating on Codalab, the files located at train_dir and test_dir will have the same d, N_0, N_1, N_2 values.
    - Please see testing1.txt, testing1_label.txt for an example.

- Prediction file: a file with N_0 + N_1 + N_2 lines
    - The value in i-th line should be your prediction of the feature vector in (i+1)-th line of test data file. Please note that you should skip the first line of test examples.
    - Your prediction should belong to {0, 1, 2}.
    - Please see testing1_label.txt as an example.

The example training and test data, as well as a sample random baseline classifier, is available on Canvas assignment page.

Linear Classifier:

You must implement a three-class linear classifier from scratch using the following method (calling third-party implementations of any classifiers is NOT allowed):
Training (using the training data set):

1. Compute the centroid of each class (0, 1, and 2)

2. Construct a discriminant function between each pair of classes (0/1, 1/2, and 2/0), halfway between the two centroids and orthogonal to the line connecting the two centroids. This is the basic linear classifier that we have discussed.

Testing (using the test data set):

1. For each test instance, use the discriminant function to decide 0 or 1 and then (depending on that answer) to decide "1 or 2" or "0 or 2". (Ties should give priority to class 0, then class 1, then class 2.

Submission Files:

You need to pack **run.py** as well as other source code into a single zip file. Please check that your run.py file provides a function named **run with 4 parameters**. You must submit this zip file on **CodaLab** for evaluation. In addition, you must attach the report about how you solve this problem to the solutions of Assignment #2, and submit them through Gradescope.

Late Submission:

Same as the homework late submission policy.

Evaluation:

The evaluation metric will be a weighted sum of the **accuracy** and **F1 score** of your predictions compared with the ground truth labels.

- **Performance** on CodaLab (30 points) = (50 * Accuracy + 50 * F1_score)*0.3
- **Reports** (10 points)
    - Report is the place for you to explain what you did, and report performance on given dataset 1 and 2.
    - Note that we will use hidden dataset 3 on CodaLab for evaluation.