Homework 2

>>JIAHUI ZHANG<<
>>908 449 6323<<

Instructions: Use this latex file as a template to develop your homework. Submit your homework on time as a single pdf file to Canvas. Please wrap your code and upload to a public GitHub repo, then attach the link below the instructions so that we can access it. You can choose any programming language (i.e. python, R, or MATLAB), as long as you implement the algorithm from scratch (e.g. do not use sklearn on questions 1 to 7 in section 2). Please check Piazza for updates about the homework. **Check https://github.com/JhZhang-1999/hw_CS760_Jiahui-Zhang for all my code and results in the folder of homework 2.**

1 A Simplified Decision Tree

You are to implement a decision-tree learner for classification. To simplify your work, this will not be a general purpose decision tree. Instead, your program can assume that

- each item has two continuous features $\mathbf{x} \in \mathbb{R}^2$
- the class label is binary and encoded as $y \in \{0, 1\}$
- data files are in plaintext with one labeled item per line, separated by whitespace:

$$x_{11} \quad x_{12} \quad y_1$$
$$\ldots$$
$$x_{n1} \quad x_{n2} \quad y_n$$

Your program should implement a decision tree learner according to the following guidelines:

- Candidate splits (j, c) for numeric features should use a threshold c in feature dimension j in the form of $x_{\cdot j} \geq c$.
- c should be on values of that dimension present in the training data; i.e. the threshold is on training points, not in between training points. You may enumerate all features, and for each feature, use all possible values for that dimension.
- You may skip those candidate splits with zero split information (i.e. the entropy of the split), and continue the enumeration.
- The left branch of such a split is the "then" branch, and the right branch is "else".
- Splits should be chosen using information gain ratio. If there is a tie you may break it arbitrarily.
- The stopping criteria (for making a node into a leaf) are that
  - the node is empty, or
  - all splits have zero gain ratio (if the entropy of the split is non-zero), or
  - the entropy of any candidates split is zero
- To simplify, whenever there is no majority class in a leaf, let it predict $y = 1$.

2 Questions

1. (Our algorithm stops at pure labels) [10 pts] If a node is not empty but contains training items with the same label, why is it guaranteed to become a leaf? Explain. You may assume that the feature values of these items are not all the same.
Solution: Yes. If a node contains training items with all the same label, the entropy value will be

$$H(Y) = -\sum_{y \in \mathcal{Y}} P(y) \log_2 P(y) = -1 \times \log_2(1) = 0 \tag{1}$$

And since the lower bound of entropy is 0 (because $P(y) \in [0, 1]$ according to the definition of probability, $\log P(y) \leq 0$, $H(Y) \geq 0$), any splitting cannot make a positive information gain. Therefore, the splitting is end here.

2. (Our algorithm is greedy) [10 pts] Handcraft a small training set where both classes are present but the algorithm refuses to split; instead it makes the root a leaf and stop; Importantly, if we were to manually force a split, the algorithm will happily continue splitting the data set further and produce a deeper tree with zero training error. You should (1) plot your training set, (2) explain why. Hint: you don't need more than a handful of items.
Solution: Consider a case shown in Table 1.

Sample No.	X1	X2	Y
1	+	-	1
2	-	+	1
3	+	+	0
4	-	-	0

Table 1: Example in Question 2-2

Here, the initial entropy is

$$H(Y) = -2 \times \frac{1}{2} \times \log_2 \frac{1}{2} = 1 \tag{2}$$

If we use X_1 to split, the sub-sample with $X_1 = +$ has an entropy of 1, and the sub-sample with $X_1 = -$ also has an entropy of 1. The situation is the same with X_2. So the algorithm would refuse to split. However, if we manually split it, say using X_1, and the next splitting in each branch will get a perfect result with zero training error.

3. (Information gain ratio exercise) [10 pts] Use the training set Druns.txt. For the root node, list all candidate cuts and their information gain ratio. If the entropy of the candidate split is zero, please list its mutual information (i.e. information gain). Hint: to get $\log_2(x)$ when your programming language may be using a different base, use `log(x)/log(2)`. Also, please follow the split rule in the first section.
Solution: See Table 2.

4. (The king of interpretability) [10 pts] Decision tree is not the most accurate classifier in general. However, it persists. This is largely due to its rumored interpretability: a data scientist can easily explain a tree to a non-data scientist. Build a tree from D3leaves.txt. Then manually convert your tree to a set of logic rules. Show the tree[1] and the rules.
Solution: My code produces the decision tree along with the splitting and predicting rules in form of plain text. Every time it reaches a leaf node and stops splitting, it prints out a line describing how this leaf node is created. Within the line, `(SPLIT:p,c)` means that the samples are split according to X_p (p is the index of features starting from 0), with threshold c. $X_p \geq c$ is split into `[LEFT]` zone, while $X_p < c$ is split into `[RIGHT]`. At the end, `(PRED:y)` means that this leaf is predicted as $Y = y$.

The result for data D3leaves.txt is shown in the following text box:

[1] When we say show the tree, we mean either the standard computer science tree view, or some crude plaintext representation of the tree – as long as you explain the format. When we say visualize the tree, we mean a plot in the 2D $\mathbf{x}$ space that shows how the tree will classify any points.

Split	$H_D(Y)$	$H_D(Y \mid S)$	$H_D(S)$	Gain Ratio
$X_1 \geq 0.1$	0.84535	0.80117	0.43950	0.10052
$X_1 \geq 0$	0.84535	0.84535	0	0
$X_2 \geq -2$	0.84535	0.84535	0	0
$X_2 \geq -1$	0.84535	0.80117	0.43950	0.10052
$X_2 \geq 0$	0.84535	0.80708	0.68404	0.05595
$X_2 \geq 1$	0.84535	0.84046	0.84535	0.00578
$X_2 \geq 2$	0.84535	0.84427	0.94566	0.00114
$X_2 \geq 3$	0.84535	0.82904	0.99403	0.01641
$X_2 \geq 4$	0.84535	0.79590	0.99403	0.04975
$X_2 \geq 5$	0.84535	0.74016	0.94566	0.11124
$X_2 \geq 6$	0.84535	0.64576	0.84535	0.23610
$X_2 \geq 7$	0.84535	0.80708	0.68404	0.05595
$X_2 \geq 8$	0.84535	0.65630	0.43950	0.43016

Table 2: Info-gain Ratio of Question 2-3

=(SPLIT:0,10.0)[LEFT]=(PRED:1)
=(SPLIT:0,10.0)[RIGHT]=(SPLIT:1,3.0)[LEFT]=(PRED:1)
=(SPLIT:0,10.0)[RIGHT]=(SPLIT:1,3.0)[RIGHT]=(PRED:0)

It means that $X_0 \geq 10$ is predicted as $Y = 1$, and for $X_0 < 10$, $X_1 \geq 3$ is predicted as $Y = 1$, and $X_1 < 3$ is predicted as $Y = 0$.

5. (Or is it?) [20 pts] For this question only, make sure you DO NOT VISUALIZE the data sets or plot your tree's decision boundary in the 2D $\mathbf{x}$ space. If your code does that, turn it off before proceeding. This is because you want to see your own reaction when trying to interpret a tree. You will get points no matter what your interpretation is. And we will ask you to visualize them in the next question anyway.

   - Build a decision tree on D1.txt. Show it to us in any format (e.g. could be a standard binary tree with nodes and arrows, and denote the rule at each leaf node; or as simple as plaintext output where each line represents a node with appropriate line number pointers to child nodes; whatever is convenient for you). Again, do not visualize the data set or the tree in the $\mathbf{x}$ input space. In real tasks you will not be able to visualize the whole high dimensional input space anyway, so we don't want you to "cheat" here.
     Solution:

=(SPLIT:1,0.201829)[LEFT]=(PRED:1)
=(SPLIT:1,0.201829)[RIGHT]=(PRED:0)

   - Look at your tree in the above format (remember, you should not visualize the 2D dataset or your tree's decision boundary) and try to interpret the decision boundary in human understandable English.
     Solution: It means whenever $X_2 \geq 0.201829$ (using natural index, starting from 1), Y is estimated to be 1, otherwise 0.
   - Build a decision tree on D2.txt. Show it to us.
     Solution: (see my Github repo for full version)

=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[LEFT]=(SPLIT:1,0.424906)[LEFT]=(PRED:1)
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[LEFT]=(SPLIT:1,0.424906)[RIGHT]=(SPLIT:0,0.70812
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[LEFT]=(SPLIT:1,0.424906)[RIGHT]=(SPLIT:0,0.70812
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[LEFT]=(SPLIT:1,0.424906)[RIGHT]=(SPLIT:0,0.70812

```
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[LEFT]=(SPLIT:1,0.424906)[RIGHT]=(SPLIT:0,0.70812
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[LEFT]=(SPLIT:1,0.424906)[RIGHT]=(SPLIT:0,0.70812
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[LEFT]=(SPLIT:1,0.424906)[RIGHT]=(SPLIT:0,0.70812
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[RIGHT]=(SPLIT:0,0.887224)[LEFT]=(SPLIT:1,0.03770
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[RIGHT]=(SPLIT:0,0.887224)[LEFT]=(SPLIT:1,0.03770
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[RIGHT]=(SPLIT:0,0.887224)[LEFT]=(SPLIT:1,0.03770
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[RIGHT]=(SPLIT:0,0.887224)[LEFT]=(SPLIT:1,0.03770
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[RIGHT]=(SPLIT:0,0.887224)[RIGHT]=(SPLIT:0,0.8503
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[RIGHT]=(SPLIT:0,0.887224)[RIGHT]=(SPLIT:0,0.8503
=(SPLIT:0,0.533076)[LEFT]=(SPLIT:1,0.228007)[RIGHT]=(SPLIT:0,0.887224)[RIGHT]=(SPLIT:0,0.8503
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[LEFT]=(SPLIT:0,0.041245)[LEFT]=(SPLIT:0,0.104043
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[LEFT]=(SPLIT:0,0.041245)[LEFT]=(SPLIT:0,0.104043
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[LEFT]=(SPLIT:0,0.041245)[LEFT]=(SPLIT:0,0.104043
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[LEFT]=(SPLIT:0,0.041245)[RIGHT]=(PRED:0)
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[LEFT]=(SPLIT:0,0.25404
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[LEFT]=(SPLIT:0,0.25404
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[LEFT]=(SPLIT:0,0.25404
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[LEFT]=(SPLIT:0,0.25404
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[LEFT]=(SPLIT:0,0.25404
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[LEFT]=(SPLIT:0,0.25404
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[RIGHT]=(SPLIT:1,0.534
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[RIGHT]=(SPLIT:1,0.534
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[RIGHT]=(SPLIT:1,0.534
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[RIGHT]=(SPLIT:1,0.534
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[RIGHT]=(SPLIT:1,0.534
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[RIGHT]=(SPLIT:1,0.534
=(SPLIT:0,0.533076)[RIGHT]=(SPLIT:1,0.88635)[RIGHT]=(SPLIT:1,0.691474)[RIGHT]=(SPLIT:1,0.534
```

- Try to interpret your D2 decision tree. Is it easy or possible to do so without visualization?

  Solution: It's not possible to interpret this result using natural language.

6. (Hypothesis space) [10 pts] For D1.txt and D2.txt, do the following separately:

   - Produce a scatter plot of the data set.
   - Visualize your decision tree's decision boundary (or decision region, or some other ways to clearly visualize how your decision tree will make decisions in the feature space).

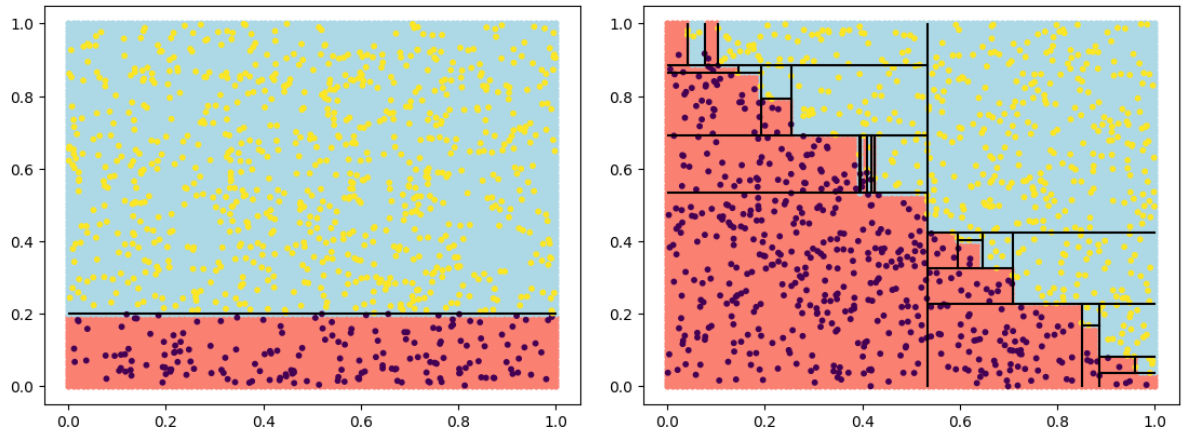
   Then discuss why the size of your decision trees on D1 and D2 differ. Relate this to the hypothesis space of our decision tree algorithm.

   Solution: See Figure 1. The black lines are decision boundaries, the red and light-blue regions denote the splitting rules, and the purple and yellow dots are the true data points.

   The difference in difficulty of interpretation between the two datasets is that, D1 is easier to be separated by fewer vertical and horizontal cuts, but in D2, the true model seems to have a linear (with slope not equal to 0 or infinity) boundary between two classes. In this case, a well-fit separation using vertical and horizontal splits only takes a huge amount of them.

7. (Learning curve) [20 pts] We provide a data set Dbig.txt with 10000 labeled items. Caution: Dbig.txt is sorted.

   - You will randomly split Dbig.txt into a candidate training set of 8192 items and a test set (the rest). Do this by generating a random permutation, and split at 8192.



(a) D1 data (b) D2 data

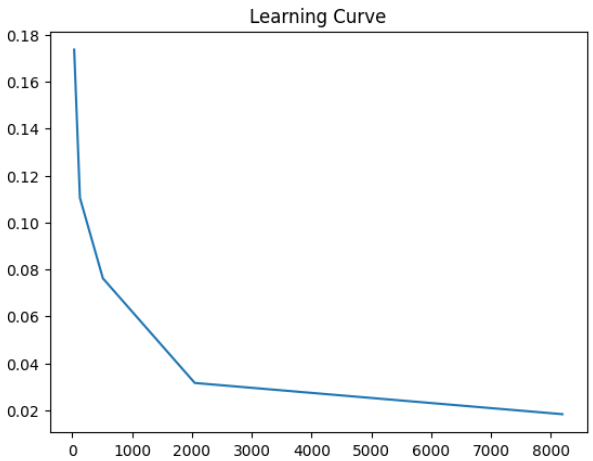
Figure 1: Scatter Plots for D1 and D2 data

- Generate a sequence of five nested training sets $D_{32} \subset D_{128} \subset D_{512} \subset D_{2048} \subset D_{8192}$ from the candidate training set. The subscript n in D_n denotes training set size. The easiest way is to take the first n items from the (same) permutation above. This sequence simulates the real world situation where you obtain more and more training data.
- For each D_n above, train a decision tree. Measure its test set error err_n. Show three things in your answer: (1) List n, number of nodes in that tree, err_n. (2) Plot n vs. err_n. This is known as a learning curve (a single plot). (3) Visualize your decision trees' decision boundary (five plots).

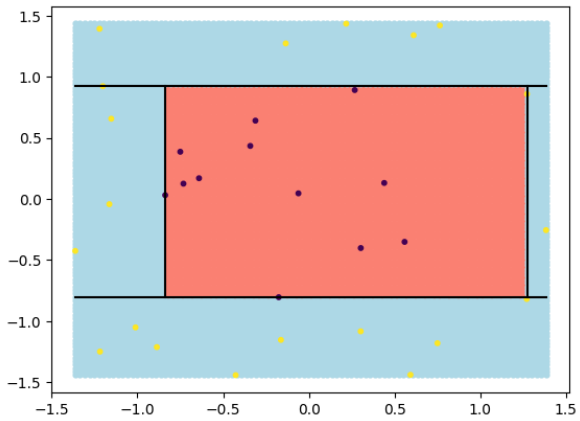
Solution: See Table 3 for the numerical results. See Figure 2 for the learning curve plot. See Figure 3 for the five scatter plots.

n (Training Sample Size)	Test Sample Size	Number of Nodes	Error Count	Error Rate
32	9968	5	1731	17.37%
128	9872	16	1091	11.05%
512	9488	33	723	7.62%
2048	7952	65	251	3.16%
8192	1808	135	33	1.83%

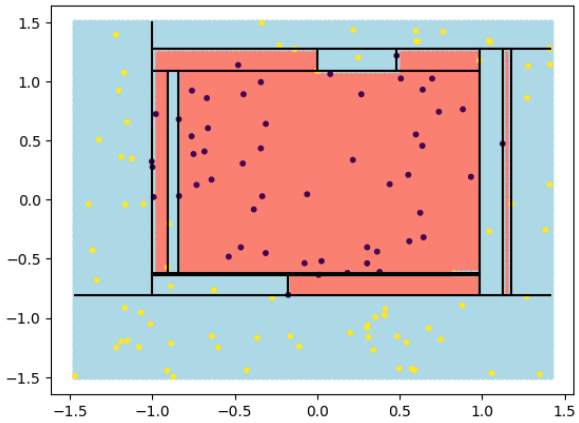
Table 3: Learning Results



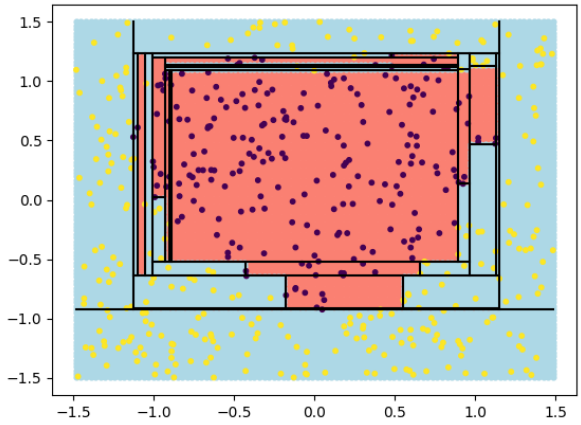
Learning Curve



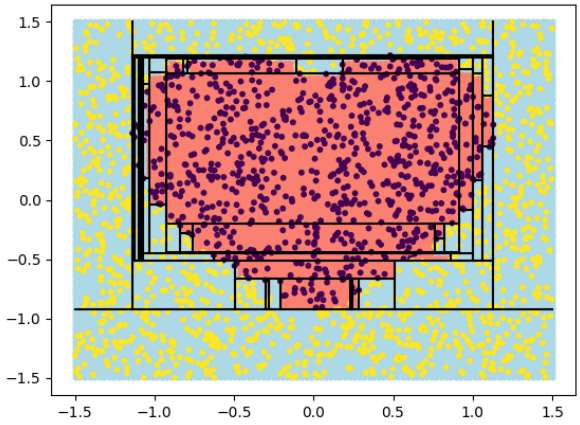
(a) D32



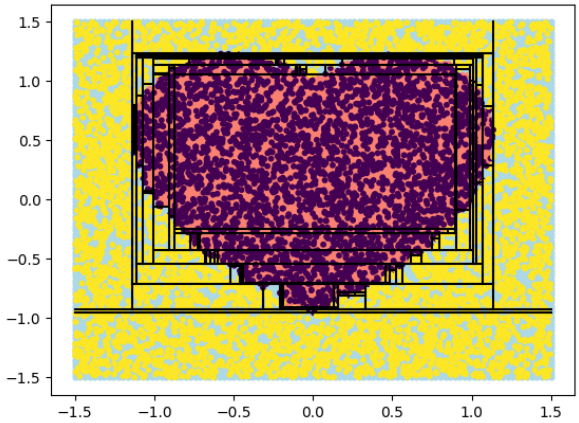
(b) D128



(c) D512



(d) D2048



(e) D8192

Figure 3: Decision Trees

3 sklearn [10 pts]

Learn to use sklearn (https://scikit-learn.org/stable/). Use sklearn.tree.DecisionTreeClassifier to produce trees for datasets $D_{32}, D_{128}, D_{512}, D_{2048}, D_{8192}$. Show two things in your answer: (1) List n, number of nodes in that tree, err_n. (2) Plot n vs. err_n.

Solution: See Table 4.

n (Training Sample Size)	Number of Nodes	Error Rate
32	7	19.40%
128	18	8.94%
512	28	5.22%
2048	56	2.29%
8192	108	1.71%

Table 4: Learning Results (using sklearn)

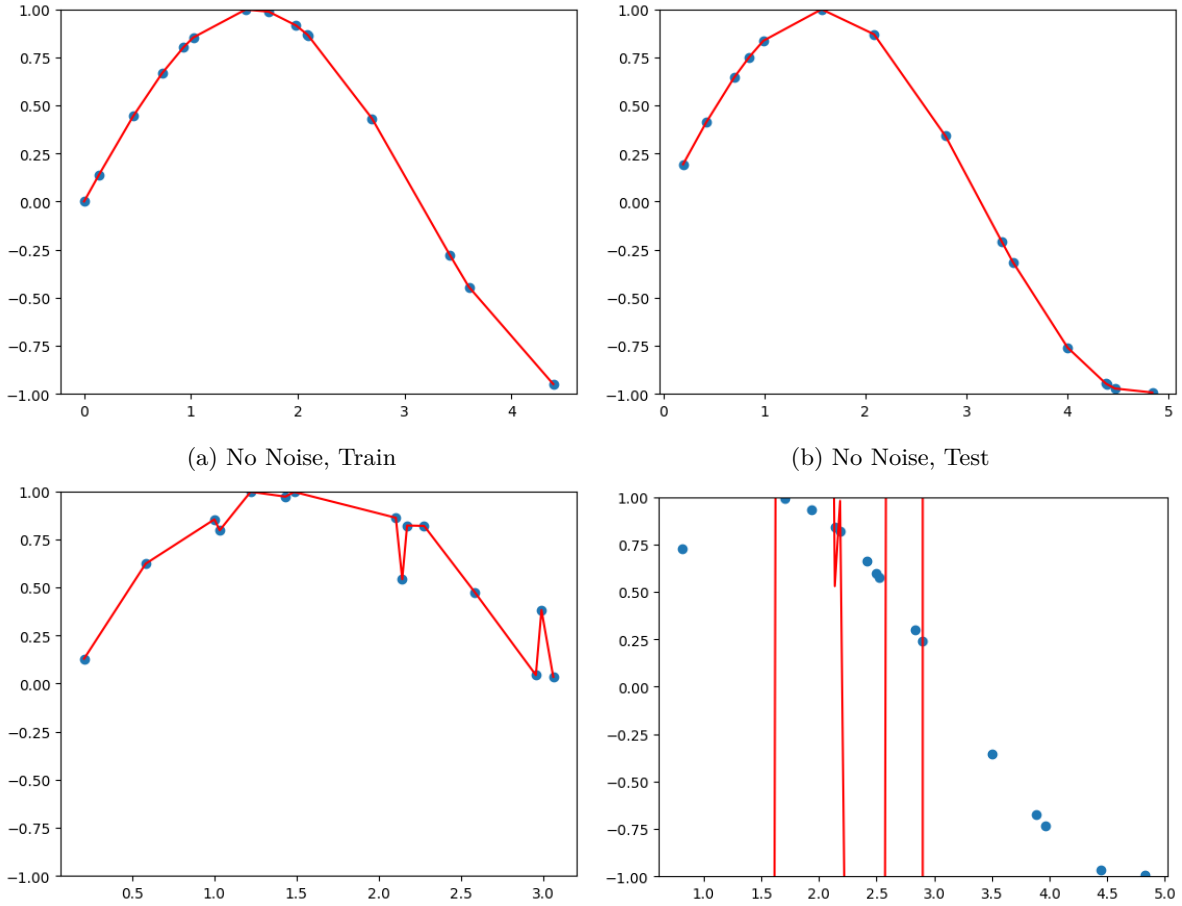
4 Lagrange Interpolation [10 pts]

Fix some interval $[a, b]$ and sample $n = 100$ points x from this interval uniformly. Use these to build a training set consisting of n pairs (x, y) by setting function $y = sin(x)$.

Build a model f by using Lagrange interpolation, check more details in https://en.wikipedia.org/wiki/Lagrange_polynomial and https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.lagrange.html.

Generate a test set using the same distribution as your test set. Compute and report the resulting model's train and test error. What do you observe? Repeat the experiment with zero-mean Gaussian noise ϵ added to x. Vary the standard deviation for ϵ and report your findings.

I used $n = 15$ since a large n leads to very unstable and bad results. In the noisy case, the random error is distributed as $\mathcal{N}(0, (\frac{1}{5})^2)$. See Figure 4 for the visualization of results. For the training set without noise, both the training and testing get good results (training MSE 8.41×10^{-9}, testing MSE 8.32×10^{-8}), while for the noisy training case, the training gets good result (MSE 1.65×10^{-7}), but the testing results are awful (MSE 9.37×10^{18}). This is because the algorithm overfits to the noisy training data, and fails to capture the real trend.



(a) No Noise, Train

(b) No Noise, Test

(c) Noisy Training, Train

(d) Noisy Training, Test

Figure 4: Lagrange Interpolation Results