

Name: _____ PUID: _____

Instructions and Policy: Each student should write up their own solutions independently. You need to indicate the names of the people you discussed a problem with; ideally you should discuss with no more than two other people.

- **YOU MUST INCLUDE YOUR NAME IN THE HOMEWORK**
- The answers (without the python scripts) **MUST** be in submitted via Gradescope.
- The python scripts will be submitted separately via turnin at data.cs.purdue.edu.
- Please write clearly and concisely - clarity and brevity will be rewarded. Refer to known facts as necessary.
- Theoretical questions **MUST** include the intermediate steps to the final answer.
- Zero points in any question where the python code answer doesn't match the answer on Gradescope.
- If the answer is a plot, it should be added to the PDF and, in the code, it should always be saved as an file (image or PDF), and **not** using `plt.show()`.

Your code is **REQUIRED** to run on Python 3 at scholar.rcac.purdue.edu. TAs will help you with the use of the scholar cluster. If the name of the executable is incorrect, it won't be graded. Please make sure you didn't use any library/source explicitly forbidden to use. If such library/source code is used, you will get 0 pt for the coding part of the assignment. If your code doesn't run on scholar.rcac.purdue.edu, then even if it compiles in another computer, your code will still be considered not-running and the respective part of the assignment will receive 0 pt.

Since in HW1, some people found it hard to create a PDF which combined different ways of writing the report, from this homework you will be allowed to submit multiple images (which should be clear) to Gradescope.

Let us restate, all of your code files should be directly put into a folder called `[your_username]_hw#`, which should not contain any subfolder. After this, you can do (1) either transfer the `[your_username]_hw#` folder to `data.cs` and compress your folder on it, or (2) compress your folder first then transfer it to `data.cs` and use `turnin` command to submit. From HW3, We will give 0 credit to those submissions that are against the requirements and will not accept regrading request.

Q0 (0pts correct answer, -1,000pts incorrect answer: (0,-1,000) pts): A correct answer to the following questions is worth 0pts. An incorrect answer is worth -1,000pts, which carries over to other homeworks and exams, and can result in an F grade in the course.

(1) Student interaction with other students / individuals:

- (a) I have copied part of my homework from another student or another person (plagiarism).
- (b) Yes, I discussed the homework with another person but came up with my own answers. Their name(s) is (are) _____
- (c) No, I did not discuss the homework with anyone

(2) On using online resources:

- (a) I have copied one of my answers directly from a website (plagiarism).
- (b) I have used online resources to help me answer this question, but I came up with my own answers (you are allowed to use online resources as long as the answer is your own). Here is a list of the websites I have used in this homework:

- (c) I have not used any online resources except the ones provided in the course website.

Theoretical Questions (8+20+14+20=62 pts)

Please submit your answers on Gradescope.

Q1 (8 pts): True or False questions

Answer the following as True or False with a justification or example. Points are uniformly distributed within the questions.

- (a) (**4 pts**) Overfitting refers to a model that can neither model the training data nor generalize well to new data.

Note: Good generalization refers to a model's ability to accurately predict new, previously unseen data.

- (b) (**4 pts**) When doing cross-validation, you need to split your dataset into training, validation and test sets.

Q2 (20 pts): Cross-validation

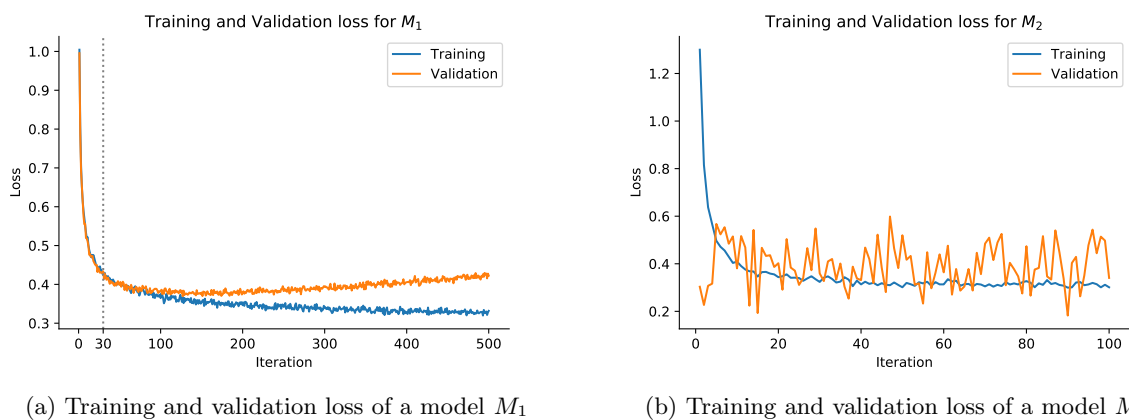


Figure 1: Example of training curves showing training and validation loss for two hypothetical models M_1 and M_2 .

1. (6 pts) Review the training and validation loss curve in Figure 1a (model M_1). The y -axis shows a loss (lower is better) for the training and validation data. The x -axis shows the iterations to update the model parameters (e.g. update steps on Perceptron algorithm or steps of gradient descent for logistic regression).

Hint: Note that these are training curves, not learning curves.

- (a) (3 pts) What are the best parameters for model M_1 ? The parameters at 500 iterations or at 100 iterations? Please justify your answer.

- (b) (3 pts) What is the disadvantage of choosing the model with parameters obtained after 30 iterations?

2. (4 pts) Review the training and validation loss curve in Figure 1b of model M_2 .

Which of the following explanations and decisions are appropriate for the observed behavior in 1b?
Choose all answers that are correct. You don't need to justify your answer.

- (A) The validation data is scarce and not very representative.
- (B) The model has not converged yet at 100 iterations and many more iterations will likely improve the training loss.
- (C) Getting more validation data will not reduce the training loss.
- (D) The training loss has converged, so obtaining more and better training data will not reduce the training loss further.



3. (4 pts) Which statement about k -fold cross validation is incorrect?

Choose all answers that are correct.

- (A) With larger k , the training set gets larger, while the test set gets smaller.
- (B) When splitting data into folds, it is desirable to minimize the variance across folds.
- (C) When using k -fold cross validation, using a smaller k means it will take more time to run the evaluation.
- (D) When $k = N$, where N is the size of the data set, k -fold cross validation is the same as leave-one-out cross validation.



4. **(6 pts)** Given the supervised learning dataset with 3 training examples $(X_1, Y_1) = (0, 1)$, $(X_2, Y_2) = (1, 1)$, $(X_3, Y_3) = (2, -1)$, use the Perceptron algorithm to learn a classifier. If needed, use $\text{sign}(0) = 1$. The Perceptron classifier will have two parameters $w = (w_0, w_1)$ (the first parameter, w_0 acts as the bias). What is the average 0-1 loss of the Perceptron algorithm evaluated by 3-fold cross validation? (Please also describe, for each fold/iteration of the cross validation, what is the training data used and the loss obtained).

Note: Without a bias term for the Perceptron, the data may not be linearly separable. As we saw in class, a simple way to add a bias term to a linear classifier is to add an extra feature with value of 1 for all data points. This way, the new data points become $X' = \{(0, 1), (1, 1), (2, 1)\}$.

Q3 (14 pts): Ensemble methods (AdaBoost)

Consider the dataset in the following table:

Index (i)	Feature 1 (X_{i1})	Feature 2 (X_{i2})	Class label (y_i)
1	-1	1	+1
2	1	1	-1
3	1	-1	+1
4	-1	-1	+1

For this question, you have to show the first few steps of the AdaBoost algorithm. The weak learner can be one of the four possible decision stumps illustrated in Fig. 2 (i.e., at each step you must choose one of the classifiers in the figure, to minimize the weighted empirical error rate, defined as COMPLETE HERE, breaking any ties by choosing the classifier with lower number).

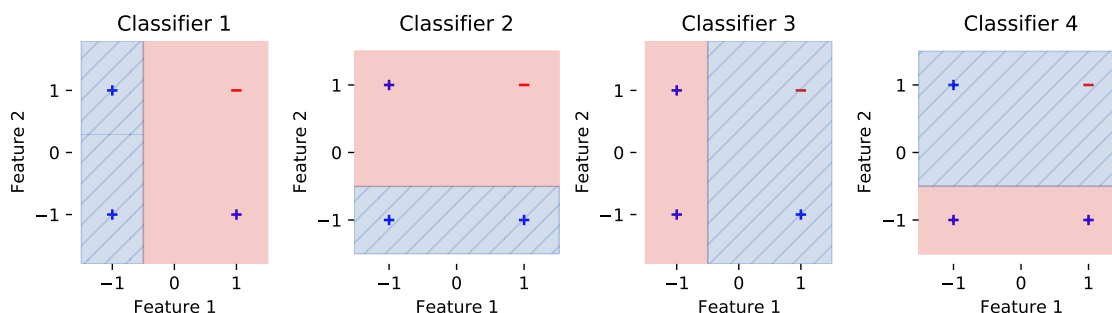


Figure 2: Choices of classifiers for the dataset. The solid red areas are assigned label -1 and the hatched blue areas are assigned label $+1$

1. (6 pts) Complete the following table, with the correct values computed during the execution of the AdaBoost algorithm, for $t = 1, \dots, 5$. Please refer to lecture slides for details of AdaBoost.

t	ξ_t	α_t	$D_t(i)$				h_t Classifier	$\mathbb{I}(h_t(i) \neq y_i)$			
			$i = 1$	$i = 2$	$i = 3$	$i = 4$		$i = 1$	$i = 2$	$i = 3$	$i = 4$
1											
2											
3											
4											
5											

2. (4 pts) Complete the following table, with the predictions computed by the AdaBoost algorithm for the training data, at the end of training (i.e. after $T = 5$ rounds).

$\hat{h}(i)$			
$i = 1$	$i = 2$	$i = 3$	$i = 4$
$\sum_{t=1}^T \alpha_t h_t(i)$			
$\hat{h}(i)$			

3. (4 pts) Assuming we stopped the algorithm after finishing the step $t = 4$, what is the predicted label for a new sample $\underline{x} = (0, -1)$? If the true label of this new test example is $+1$, what is the 0-1 loss for this example?

Q4 (20 pts): SVM

Consider an SVM that obtains a decision boundary through the maximization optimization in Equation (1), which is equivalent to the minimization optimization in Equation (2), where β and β_0 denote the parameters, x_i the i -th training example, $y_i \in \{-1, 1\}$ the class label for the i -th training example, M the margin, N the number of instances.

$$\begin{aligned} & \max_{\beta, \beta_0, \|\beta\|_2=1} M \\ \text{subject to } & y_i (x_i^T \beta + \beta_0) \geq M, \quad i = 1, \dots, N. \end{aligned} \tag{1}$$

which is equivalent to

$$\begin{aligned} & \min_{\beta, \beta_0} \frac{1}{2} \|\beta\|_2^2 \\ \text{subject to } & y_i (x_i^T \beta + \beta_0) \geq 1, \quad i = 1, \dots, N. \end{aligned} \tag{2}$$

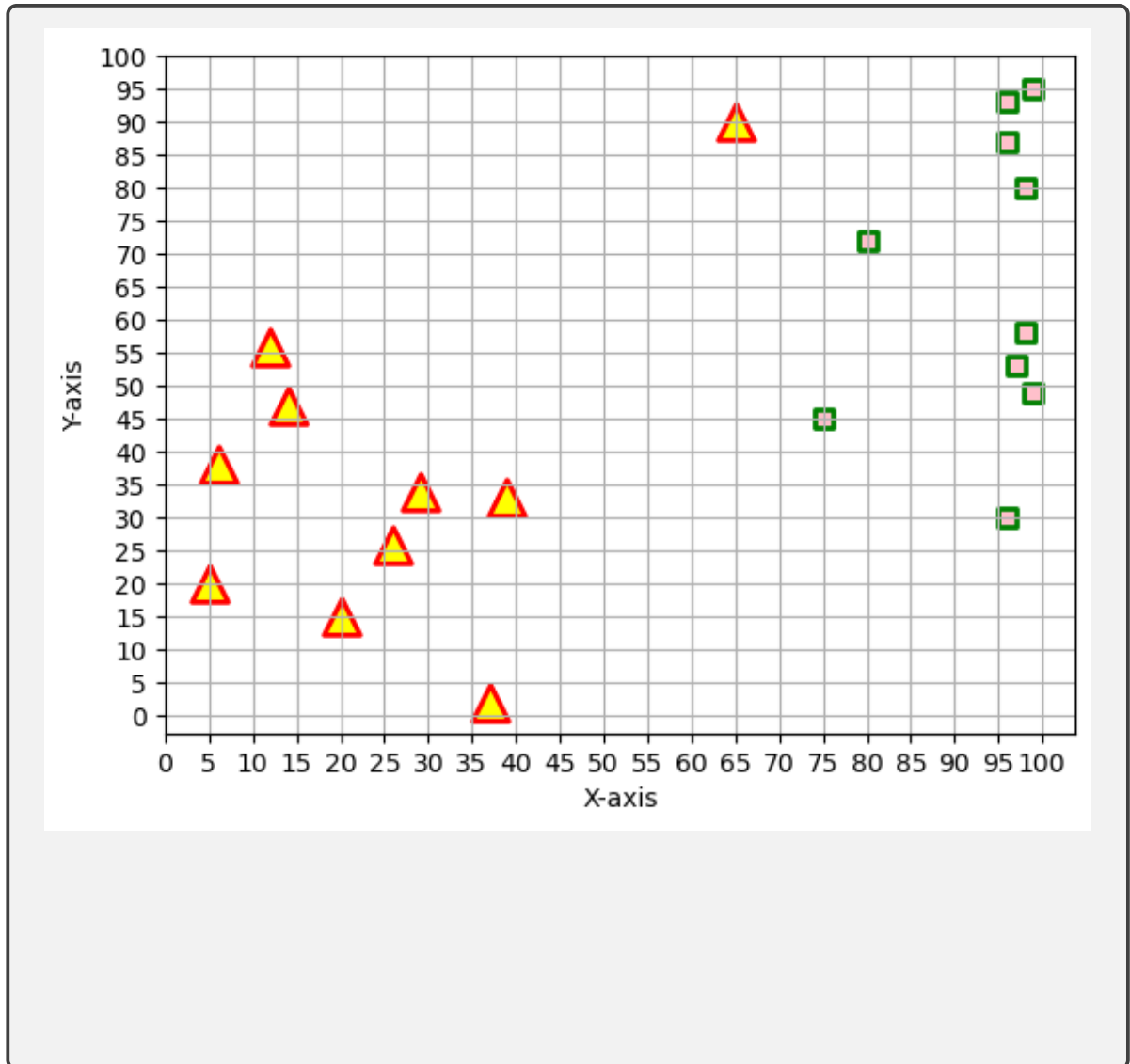
Mark true or false for the following assertions (Q4.1, Q4.2, Q4.3) and justify your answer. Questions with no justification will receive 0 points.

1. **(4 pts)** This SVM objective assumes data is linearly separable.

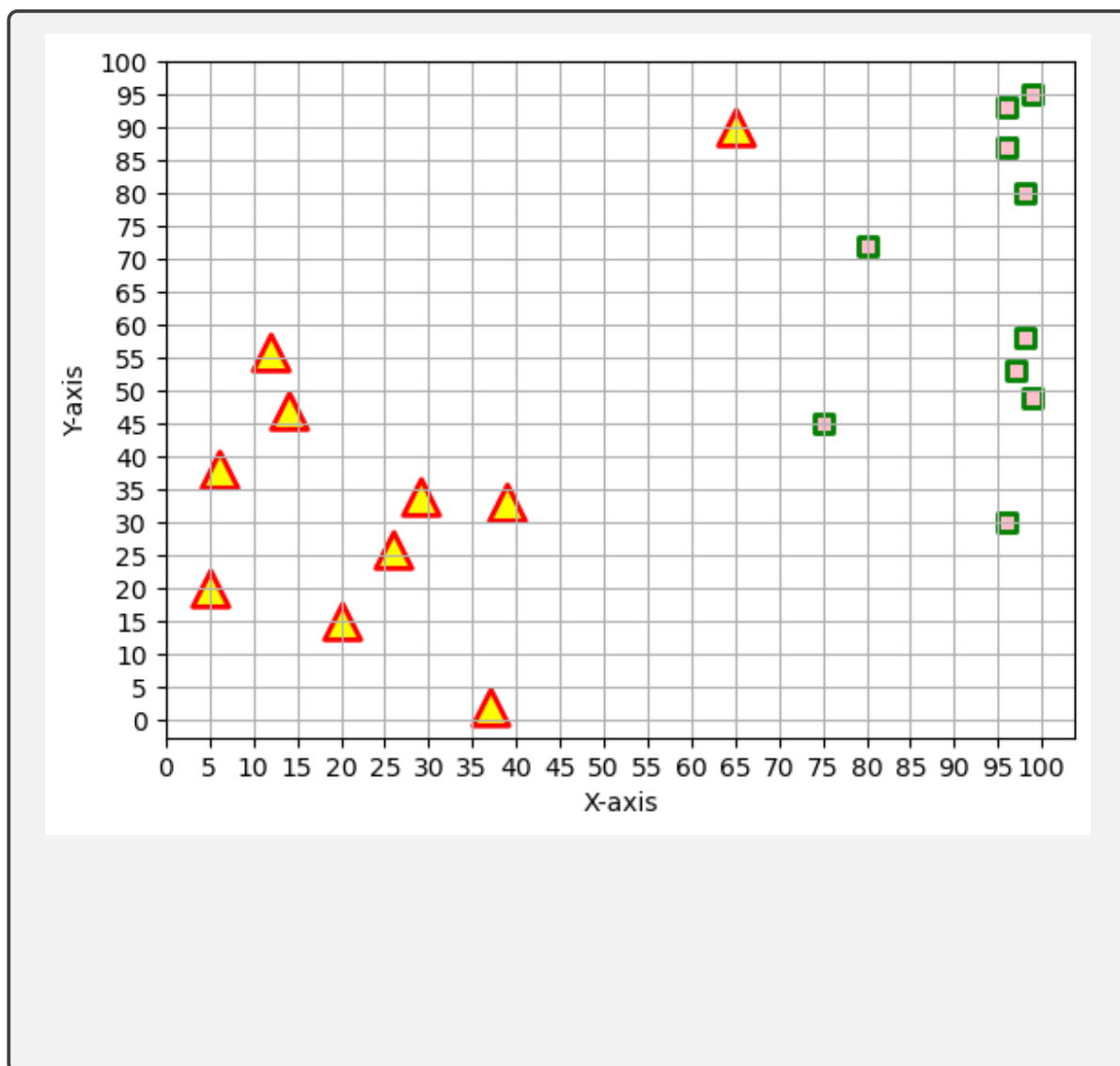
2. **(4 pts)** After obtaining the decision boundary in Equation 1, if we modify β while maintaining the restriction $\|\beta\|_2^2 = 1$, this would modify the minimum distance from the decision boundary to the origin.

3. **(4 pts)** Assume the maximum margin obtained by the model was $M = 4 = d_1 + d_2$, where $d_1 = 2$ and $d_2 = 2$ denote the distance from the decision boundary to hyperplanes H_1 and H_2 defined by the support vectors. If we add 1 to β_0 , the new maximum margin would be $M + 1$.

4. (4 pts) **SVM decision boundary.** Consider again an SVM whose decision boundary is obtained by Equations 1 and 2 like in the previous questions. Now, consider the training dataset with two training classes (signaled as squares and triangles) given by the scatter plot in the following figure. Draw the decision boundary for this dataset obtained by our SVM on it. The drawing needs to correctly separate the examples but there are multiple possible solutions. Each point of the line you draw must match the points of the true decision boundary within the error margin of 10 unit intervals along the x -axis and y -axis.



5. (4 pts) **SVM decision boundary with removed example.** Now pick a single data point \mathbf{x}_i that, if removed, would yield the maximum possible increase in the margin of a new SVM decision boundary over the dataset without \mathbf{x}_i . Indicate which point is \mathbf{x}_i in the figure below (reproduction of last question). Draw the new decision boundary after \mathbf{x}_i 's removal on it. The drawing needs to correctly separate the remaining examples but there are multiple possible solutions. Each point of the line you draw must match the points of the true new decision boundary within the error margin of 10 unit intervals along the x -axis and y -axis.



Programming Part (20+18=38 pts)

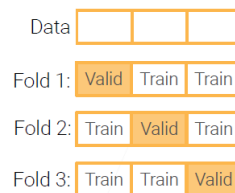
There are some basic requirements that you should follow when you submit your code:

- Make sure you don't import any package that you are allowed.
- Make sure your returned data type follows the given data type.
- Make sure your code does not contain `matplotlib` popups, such as `plt.show()`.

Q5 (20 pts): Cross-Validation

In this assignment, you will use a k -fold cross validation technique to measure the performances and tune hyperparameters of some algorithms that you have learned or implemented, including KNN, Decision Tree, Logistic Regression and SVM. You will still be using the dating dataset that you have seen in HW2. Again, it is a part of the whole dataset, while the test data is not published to you and will be used for grading. Please refer to the previous homework if you have questions about datasets.

A k -fold cross-validation is implemented as follows.



- Partition the training data into k parts.
- For $i = 1, \dots, k$: use the i -th part as the validation set and the rest for training
- Report the validation error averaged over k rounds.

(Huang, Li and Smola, 2021)

1. Implementation

Implement the required functions in the file `cross_validation.py`:

- (a) **(3 pts)** Implement `create_folds()` method, which takes in data samples \mathbf{X} and their labels \mathbf{y} . You should partition the data into k -folds and return a list of tuples of (\mathbf{X}, \mathbf{y}) with each element in the tuple being a fold. Please refer to the code comments for a sample output.
- (b) **(3 pts)** Implement `train_valid_split()` method, which takes in a list of folds and an integer index at which the fold will be used as the validation set. Combine the rest of the folds to get the training set. Return two tuples: $(\mathbf{X}_{\text{train}}, \mathbf{y}_{\text{train}})$, $(\mathbf{X}_{\text{val}}, \mathbf{y}_{\text{val}})$. Please refer to the code comments for a sample output.
- (c) **(3 pts)** Implement `cross_val_score()` method, which takes in data samples \mathbf{X} , their labels \mathbf{y} , and a classifier `clf` as input, and should return two lists: `train_accs` and `val_accs`, where each element represents the accuracy of the training or validation dataset of a specific folds split case. Please refer to the code comments for a sample output.

Implement the required functions in the file `evaluation.py`:

- (d) **(3 pts)** Implement `find_best_param` method, which uses k -fold cross validation to find the best parameter of a model. Models can be Logistic Regression, Decision Tree and SVM. It takes the pre-processed dataset (X, y) , the number of folds k , the model name (`'logistic'`, `'decision_tree'` or `'svm'`), and the corresponding parameters as input, and should return the training and validation accuracy dictionaries and also the best parameter.
- For Logistic Regression model, you will be using the previous code and tune the hyperparameter `lr` (learning rate).
 - For Decision Tree, you will be using the sklearn library, `sklearn.tree.DecisionTreeClassifier`, and tune the hyperparameter `max_depth`.
 - For SVM, you will be using the sklearn library `sklearn.svm.LinearSVC`, which is a linear SVM, and tune the hyperparameter C , which is a regularization parameter, and basically determines the influence of misclassification on the data samples. The larger the value of C is, the less tolerant your model will be to misclassified samples.

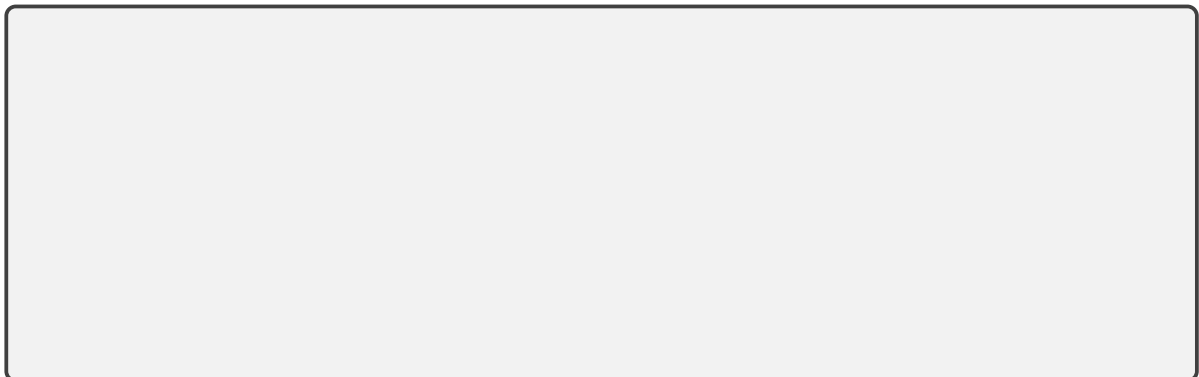
Please refer to the code comments for a sample output.

2. Evaluation

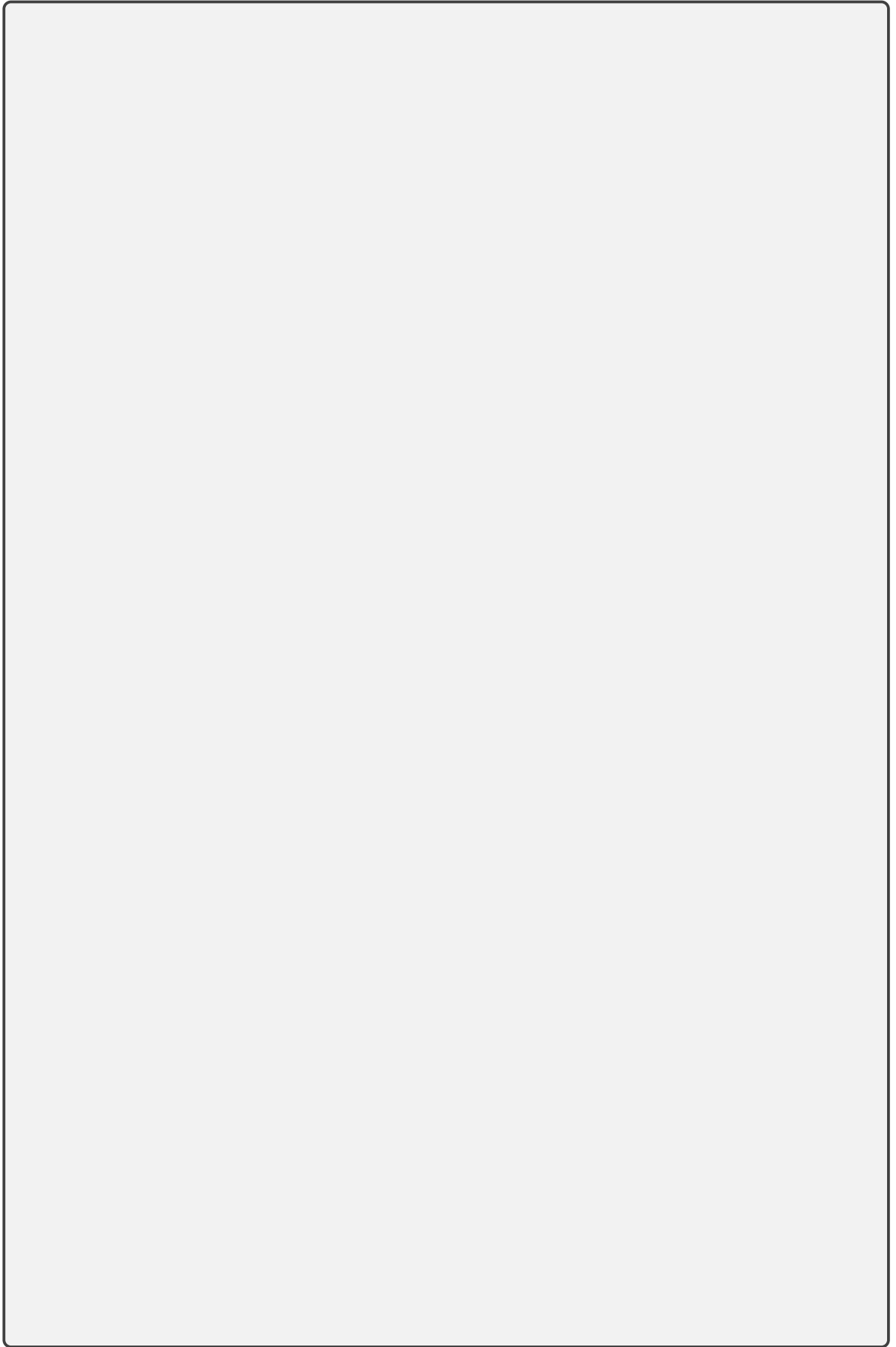
This part is simple. We have implemented many functions for you, including the plotting of learning curves, cross validation processes and the main function. Please refer to the code for more details. If your implementations are correct, you will be able to get all the results. What you need to do is just attaching your results to the final report.

- (a) **(2 pts)** Print out your best parameters for the three models that you tuned by using k -fold cross validation. The format of your output should be:

```
The best learning rate for Logistic Regression is ...
The best max_depth for Decision Tree is ...
The best C for SVM is ...
```



- (b) **(6 pts)** Attach the learning curves of Logistic Regression, DecisionTree, and SVM in your report, which can be automatically generated under your working directory (3 figures). Briefly discuss your observations.



Q6 (18 pts): Bagging and Boosting

Bagging is generally used to help stabilize classifiers with unstable learning algorithms (optimal score searching algorithms). A classifier has a stable learning algorithm if, by changing the training data, the predicted class labels in the test data don't change. For instance, the predictions of a decision tree might significantly change with a small change in the training data. This definition depends on the amount of data, of course. Classifiers that are unstable with 10^3 training examples may be stable with 10^9 examples. Bagging works by aggregating the answers of unstable classifiers trained over multiple training datasets. These multiple datasets are often not independent, generally sampled with replacement from the same training data.

Boosting works by converting weak classifier (very simple models) to strong ones (models that can describe complex relationships between the inputs and the class labels). A weak learner is a classifier whose output of an test example attributes x_i is only slightly correlated with its true class t_i . That is, the weak learner classifies the data better than random, but not much better than random. In boosting, weak learners are trained sequentially in a way that the current learner gives more emphasis to the examples that past learners made mistakes on.

1. **(8 pts)** Suppose we decide to use a large deep feedforward network as a classifier with a small training dataset. Assume the network can perfectly fit the training data but we want to make sure it is accurate in our test data (without having access to the test data). Would you use boosting or bagging to help improve the classification accuracy? Describe what would be the problem of using the other approach.

2. **(10 pts)** Download the code at `mystery_classifier.zip` . Read the code and answer the following questions:

Note: You don't need to run the code, but if you want to run it on the `scholar.rcac.purdue.edu` cluster, you will need to command `module load anaconda/5.1.0-py36`

- (a) **(5 pts)** Which classifier is this? Why are the trees used in this classifier so shallow? Describe how the classifier works using pseudo-code.

Hint: It uses decision trees but this is not a decision tree classifier.

- (b) **(5 pts)** What happens if we have mislabeled data? Why mislabeled data could be a problem?

Hint: We are looking for an answer that uses the training weights of the examples at each iteration as a justification.

Submission Instructions

Please read the instructions carefully. Failing to follow any part might incur some score deductions.

1. PDF upload

Your report should be **one** a PDF file or multiple image files. Please only upload multiple image files if you are very familiar with Gradescope.

Important: Associate the Gradescope questions with your upload no matter it is a PDF or multiple images. TAs have the right to give zero points if it is not clear where the answer to a question is because it was not linked on Gradescope.

As stated at the beginning, please remember that **clarity and conciseness** will be rewarded.

There are some suggestions (but not limited to) that you can write the PDF report:

- 1) Print out the PDF, use your pens to write in real paper, scan or take photos, combine these photos into a MS Word file, and print as PDF. It is recommended to scan using some apps, like *Notes* on iOS and *Adobe Scan* on Android, because this will make images clearer than solely taking photos. (Of course if you want to use a scanner).
- 2) Use electronic devices (like iPad) and pens (like Apple pencil), write your solutions in the given grey colored text boxes, and export to PDF. If you are very familiar with Gradescope, you can try to upload pictures directly.
- 3) Write your answers in the given `.tex` file and export to PDF.

If you decide to upload multiple images, **please scan them before uploading** (check suggestion 1) above), because photos are not easy to grade. **TAs have the right to give zero points if your submission is not readable.**

The report PDF must be **uploaded on Gradescope** (see link at Brightspace)

2. Code Upload

Naming convention: `[your_username]_hw3`

All your submitting code files should be included in one folder. The folder should be named with the above naming convention. For example, if my username is “smith”, then for Homework 3, my folder name should be “smith_hw3”.

The **folder structure** should be **exactly the same** as the following directory tree:

```
smith_hw3
├── cross_validation.py
├── dating_train.csv
├── evaluation.py
├── logistic_regression.py
└── utils.py
```

It is ok if you include your plotted figures in your folder.

We noticed that there are some unqualified submissions of previous homework assignments:

- Some of you put all your code files into the original `hw#.code` folder that you were given. Some even included the `hw#_handout` folder which contains the PDF and latex handouts.
- Some of you used PyCharm to develop your code and included the `venv` folder in your submission. This made your submission extremely large.

These cases made it hard for our autograding script hard to find your code and had to give your code 0 point. Let us restate, all of your code files should be directly put into a folder called `[your_username]_hw#`, which should not contain any subfolder. After this, you can do (1) either transfer the `[your_username]_hw#` folder to `data.cs` and compress your folder on it, or (2) compress your folder first then transfer it to `data.cs` and use `turnin` command to submit. From HW3, We will give 0 credit to those submissions that are against the requirements and will not accept regrading request.

Use the following command to compress your folder:

```
tar -czvf [your_username]_hw3.tar.gz [your_username]_hw3
```

Submit the code: TURNIN Instructions

Please submit your compressed file on `data.cs.purdue.edu` by `turnin` command line, e.g.

```
turnin -c cs373 -p HW3 [your_username]_hw3.tar.gz
```

Please check the following notes before you submit your code:

- Please make sure you didn't use any library/source explicitly forbidden to use. If such library/source code is used, you will get 0 pt for the coding part of the assignment.
- If your code doesn't run on `scholar.rcac.purdue.edu`, then even if it compiles in another computer, your code will still be considered not-running and the respective part of the assignment will receive 0 pt.