CS37300 Homework	— Homework 3—	Last update: Oct 15, 2025
Due Date: Oct 29, 2025 (11:59 pm)		

Name: Luis Castellanos PUID: 33489855

Instructions and Policy: You are allowed to study with others and use online resources for reference, however, the work you turn in must be your own. This means do not copy/paste from stackexchange (or from another student.) If you have worked closely with other students, provide their name(s) and a brief (at most one paragraph) description of the interaction; if we feel this oversteps the bounds we will discuss it with you. Each student should write up their own solutions independently.

The requirements below are supposed to be followed in this and further homework assignments.

- 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). Please remove `plt.show()` in your code, because it will interrupt our grading process.

Your code is REQUIRED to run on Python 3 at scholar.rcac.purdue.edu. The TA's will help you with the use of the scholar cluster. 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.

Theoretical Questions (8+20+14+20+18=80 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 wihtin the questions.

(a) (**4 pts**) High bias can result in the model learning random noise in the training data.

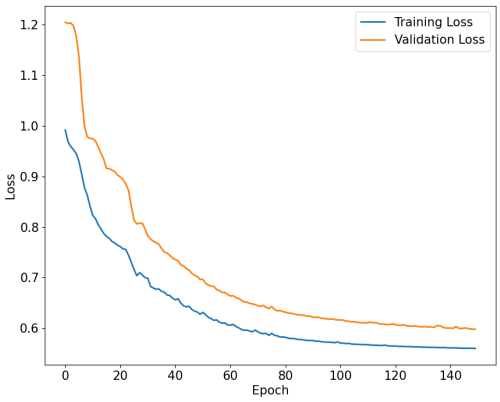
False, high bias can not result in the model overfitting the data.

(b) (**4 pts**)

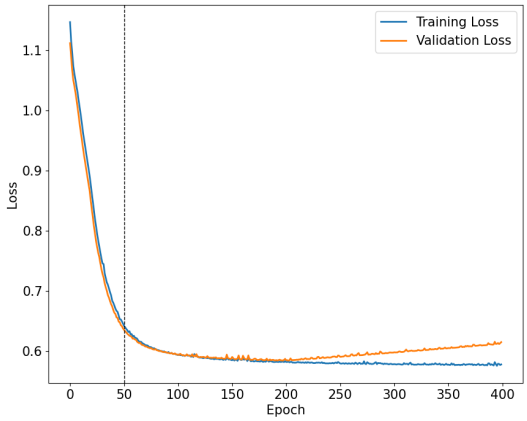
A Gaussian Process can approximate any function more flexibly than a Generalized Linear Model, given a suitable choice of kernel.

True, given a suitable choice of kernel GP are able to captur complex patterns and relationships better than GLMs.

Q2 (20 pts): Cross-validation



(a) Training and validation loss of a model M_1



(b) Training and validation loss of a model M_2

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 1b (model M_2). 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_2? The parameters at 150 iterations or at 400 iterations? Please justify your answer.

   Parameters at 150 iterations are best for model M_2 because parameters at 400 seem like overfit the data. Also validation loss(150) $<$ validation loss(400) and we want the smaller of the two.

(b) (**3 pts)** What is the disadvantage of choosing the model with parameters obtained after 50 iterations?

The model could underfit the data, training and validation loss are still high, and possibility that the model hasn't trained enough.

2. **(4 pts)** Review the training and validation loss curve in Figure 1a of model M_1.

   Which of the following explanations and decisions are appropriate for the observed behavior in 1a?
   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 training data does not provide sufficient information to learn the problem relative to the validation data used to evaluate it.

   (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.

   (B),(C)

3. **(4 pts)** Which statement about k-fold cross-validation is incorrect?
   Find out the false statements.

   (A) When using k-fold cross-validation, using a smaller k means it will take more time to run the evaluation.

   (B) With larger k, the training set gets larger, while the test set gets smaller.

   (C) When splitting data into folds, it is desirable to minimize the variance across folds.

   (D) When $k = N$, where N is the size of the data set, k-fold cross-validation is not the same as leave-one-out cross-validation.

   (A),(C),(D)

4. **(6 pts)** Given the supervised learning dataset with 3 training examples $(X_1, Y_1) = (-2, -1)$, $(X_2, Y_2) = (-1, -1)$, $(X_3, Y_3) = (0, 1)$, use the Perceptron algorithm to learn a classifier. If needed, use 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).

Test: (-2, -1) Train: (-1,-1),(0,1):

(1) $\hat{y} = sign(0 + (0)(-1)) = sign(0) = 1 \neq y$, update $w = (-1, 1)$
$\hat{y} = sign(-1 + (1)(0)) = sign(-1) = -1 \neq y$, update $w = (0, 1)$

(2) $\hat{y} = sign(0 + (1)(-1)) = sign(-1) = -1 = y$
$\hat{y} = sign(0 + (1)(0)) = sign(0) = 1 = y$

Validation: $\hat{y} = sign(0 + (1)(-2)) = sign(-2) = -1 = y$, Loss = 0

Test: (-1, -1) Train: (-2,-1),(0,1):

(1) $\hat{y} = sign(0 + (0)(-2)) = sign(0) = 1 \neq y$, update $w = (-1, 2)$
$\hat{y} = sign(-1 + (2)(0)) = sign(-1) = -1 \neq y$, update $w = (0, 2)$

(2) $\hat{y} = sign(0 + (2)(-2)) = sign(-4) = -1 = y$
$\hat{y} = sign(0 + (2)(0)) = sign(0) = 1 = y$

Validation: $\hat{y} = sign(0 + (2)(-1)) = sign(-2) = -1 = y$, Loss = 0

Test: (0, 1) Train: (-2,-1),(-1,-1):

(1) $\hat{y} = sign(0 + (0)(-2)) = sign(0) = 1 \neq y = -1$, update $w = (-1, 2)$
$\hat{y} = sign(-1 + (2)(-1)) = sign(-3) = -1 = y$

(2) $\hat{y} = sign(-1 + (2)(-2)) = sign(-5) = -1 = y$
$\hat{y} = sign(-1 + (2)(-1)) = sign(-3) = -1 = y$

Validation: $\hat{y} = sign(-1 + (2)(0)) = sign(-1) = -1 \neq y$, Loss = 1

Avg Loss = $\dfrac{0 + 0 + 1}{3} = \dfrac{1}{3}$

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 three 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 $\varepsilon_t = \sum_{i=1}^{n} D_t(i)\mathbb{I}[h_t(x_i) \neq y_i]$, where t means the number of iterations and $D_t(i)$ means the reweighting of data at round t, breaking any ties by choosing the classifier with lower number). You may refer to this link `https://drive.google.com/file/d/1OWABdO_gvkOcDgt3WsiNx_b-hoOUOvXM/view?usp=sharing` for details of AdaBoost.

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. **(3 pts)** Start with the first iteration of the AdaBoost algorithm, where the initial distribution is uniform over the training examples. Fill in the first row of the table to get familiar with the algorithm (with detailed calulation steps).

t	$D_t(i)$ $i=1$	$i=2$	$i=3$	$i=4$	h_t Classifier	$\mathbb{I}(h_t(x_i) \neq y_i)$ $i=1$	$i=2$	$i=3$	$i=4$	ε_t	α_t
1	1/4	1/4	1/4	1/4	1	0	1	0	0	1/4	0.5493

Classifierh_i	$-$	$+$
h_1	$x_2 > 0$	$x_2 < 0$
h_2	$x_1 < 0$	$x_1 > 0$
h_3	$x_2 < 0$	$x_2 > 0$

$h_1 \rightarrow$ one wrong sample 2, $h_2 \rightarrow$ one wrong sample 4, $h_1 \rightarrow$ three wrong samples 1,3,4
so h_1 and h_2 are tied with $\epsilon = 1/4$ pick h_1
$\alpha_1 = \frac{1}{2} ln(\frac{1-1/4}{1/4}) \approx 0.5493$

2. **(3 pts)** Repeat the calculations for the next two iterations of the AdaBoost algorithm and fill in the table below.

	$D_t(i)$				h_t	$\mathbb{I}(h_t(x_i) \neq y_i)$					
t	$i=1$	$i=2$	$i=3$	$i=4$	Classifier	$i=1$	$i=2$	$i=3$	$i=4$	ε_t	α_t
2	0.166	1/2	0.166	0.166	2	0	0	0	1	0.166	0.807
3	0.1	0.3	0.1	0.5	1	0	1	0	0	0.3	0.4236

i	y_i	$h_2(x_i)$	$e^{\pm\alpha_1}$	$D_1(i)$	$D_2(i)$	$D_2(i)/(Z_1 = 0.866)$
1	-1	-1	$e^{-0.5493}$	1/4	0.144	0.166
2	$+1$	-1	$e^{+0.5493}$	1/4	0.433	1/2
3	$+1$	$+1$	$e^{-0.5493}$	1/4	0.144	0.166
4	$+1$	$+1$	$e^{-0.5493}$	1/4	0.144	0.166

Pick h_2 with smalles $\epsilon = 0.166$

i	y_i	$h_1(x_i)$	$e^{\pm\alpha_2}$	$D_2(i)$	$D_3(i)$	$D_3(i)/(Z_1 = 0.754)$
1	-1	-1	$e^{-0.807}$	0.144	0.0754	0.1
2	$+1$	-1	$e^{-0.807}$	0.433	0.2262	0.3
3	$+1$	$+1$	$e^{-0.807}$	0.144	0.0754	0.1
4	$+1$	$+1$	$e^{+0.807}$	0.144	0.377	0.5

Pick h_1 with smalles $\epsilon = 0.3$

3. **(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 $T = 3$ rounds).

	$\hat{h}(x_i)$			
	$i=1$	$i=2$	$i=3$	$i=4$
$\sum_{t=1}^{T} \alpha_t h_t(x_i)$	-1.776	-0.1682	+1.7776	0.1682
$\hat{h}(x_i)$	-1	-1	+1	+1

t	h_t	ϵ_t	α_t
1	h_1	0.25	0.5493
2	h_2	0.166	0.8047
3	h_3	0.3	0.4236

$\hat{h}(x) = sign(h_1(\alpha_1 + \alpha_3) + h_2\alpha_2) = 0.9729(h_1(x)) + 0.8047(h_2(x))$

4. **(4 pts)** Assuming we stopped the algorithm after finishing the step $t = 2$, 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?

$h_1(x) = 1$ and $h_2(x) = +1$

$(0.5493)(-1) + (0.8047)(+1) = 0.2554 \Rightarrow +1$

$\hat{h}(x) = +1$ and $y = 1$ so Zero One Loss $= 0$

Q4 (20 pts): SVM

Consider an SVM that obtains a linear decision boundary through finding a hyperplane indexed by $(\hat{\mathbf{w}}, \hat{b})$ that is consistent with the training examples while maximizing the margin with respect to the training examples as described in Equation (1), which is equivalent to the quadratic optimization problem described in Equation (2), where $(\mathbf{w}, b)$ denote the parameters of the hyperplane, x_i the i-th training example, $y_i \in \{-1, 1\}$ the class label for the i-th training example, N the number of instances.

$$(\hat{\mathbf{w}}, \hat{b}) = \arg \max_{(\mathbf{w},b):\|\mathbf{w}\|=1} \min_{1 \le i \le n} y_i(\mathbf{w}^T x_i + b) \tag{1}$$

which is equivalent to

$$\begin{aligned} &\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|_2^2 \\ &\text{subject to } \; y_i(\mathbf{w}^T x_i + b) \ge 1, \; i = 1, \ldots, N. \end{aligned} \tag{2}$$

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 works for both linearly separable and non-linearly-separable data.

   False by equations (1) and (2) this is hard-margin SVM which requires that the data is linearly separable.

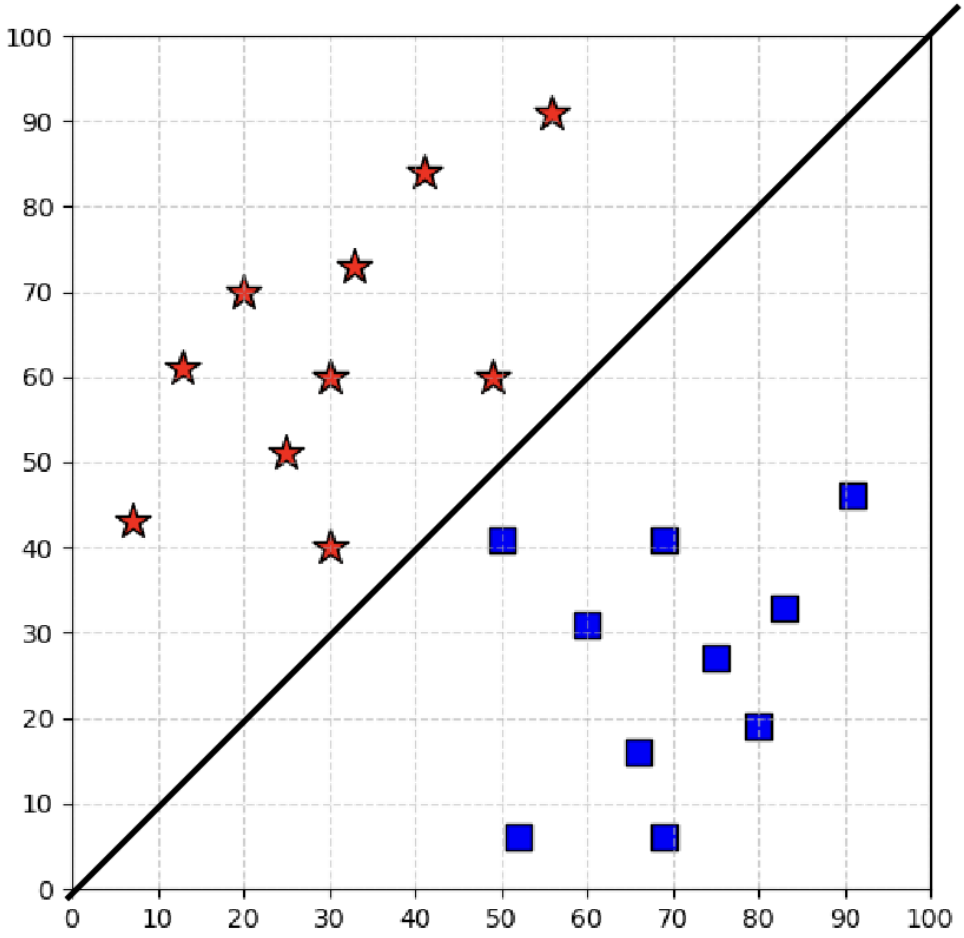
2. **(4 pts)** After obtaining the decision boundary in Equation 1, if we modify $\mathbf{w}$ while maintaining the restriction $\|\mathbf{w}\|_2^2 = 1$, this would modify the minimum distance from the decision boundary to the origin.

   False, since the distance from the origin to the decision boundary is $\frac{|b|}{||w||}$ any change in w generally changes b as well, so the distance to the boundary to the origin will change.

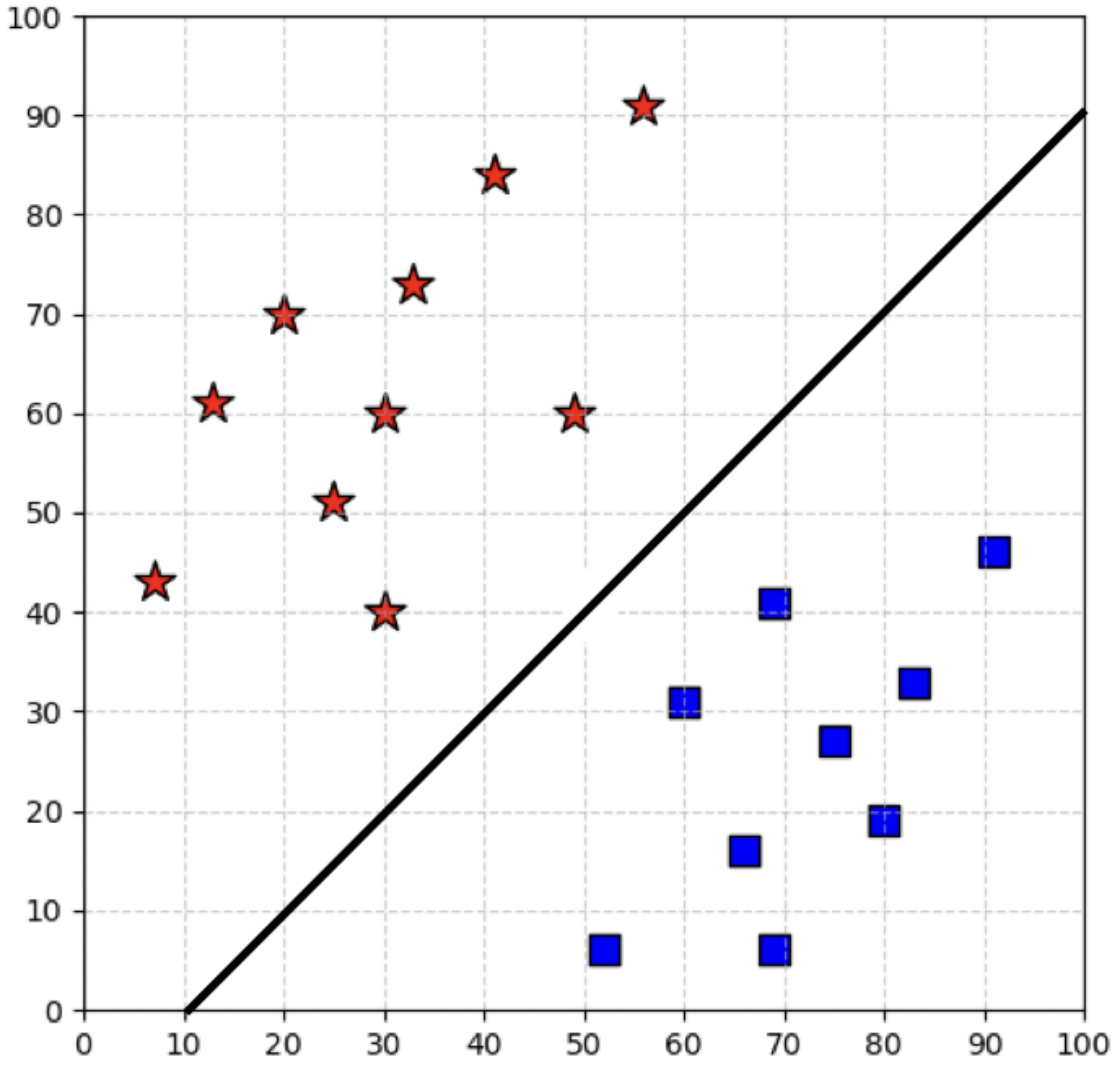
3. **(4 pts)** The distances from the optimal separating hyperplane $(\hat{\mathbf{w}}, \hat{b})$ to the closest point within both classes are the same. In other words, the optimal hyperplane must lie right in the middle of the support vectors belonging to the two classes.

   True, in the hard-margin SVM the closest points from each class to decision boundary lie on hyperplanes. Each hyperplane is exactly $1/||w||$ units from the decision boundary, so SVM solution bisects the support vector.

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 stars) 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 could be multiple possible solutions. Each point of the line you draw must match the points of the true decision boundary within the error margin of 5 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 could be 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 5 unit intervals along the x-axis and y-axis.



Q5 (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.

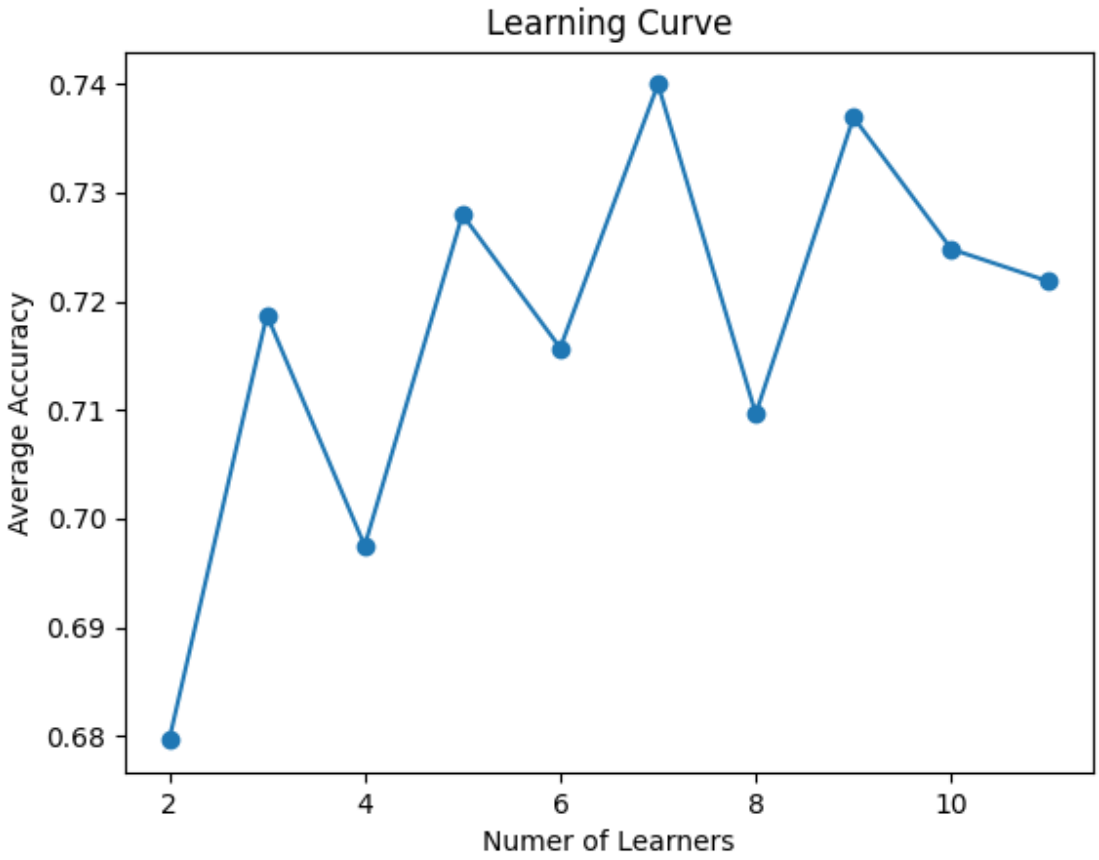
Boosting works by converting weak classifier (very simple models) to strong ones (models that can describe complex relationships between the inputs and the class labels). A weak learner is a classifier whose output of an test example attributes x_i is only slightly correlated with its true class t_i. 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 learns made mistakes on.

1. **(9 pts)** Suppose we decide to use an SVM classifier with a small training dataset. Assume the SVM model 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.

   Use bagging with small and a perfect SVM the main risk is variance, so bagging can help smooth out predicitons by averaging over re-samples. Using boosting will lead to overfitting because the sVM already fits the data perfectly

2. **(9 pts)** Suppose we run a decision tree classifier on a large training dataset. The classifier is unable to fit the training data perfectly. Would you use boosting or bagging to help improve the classification accuracy? Describe what would be the problem of using the other approach.

Use boosting a single decision tree may underfit and boosting improves the tree with iterations by focusing on errors of bias. Meanwhile bagging can reduce variance but it won't alleviate the tree's underfitting like boosting.


Learning Curve
Average Accuracy
0.74
0.73
0.72
0.71
0.70
0.69
0.68
2
4
6
8
10
Numer of Learners