Testing/Validation and Ensemble Methods Quiz

1. **Training/Validation/Testing**

   (a) Describe the problem with using a test set for both model evaluation and hyperparameter tuning.

   (b) You and your friend are arguing over whether to use 3-fold or 20-fold cross-validation to tune the hyperparameters of your learning algorithm. What arguments might be made for the two alternatives?

   (c) As an analyst at a car company, you are attempting to build a machine learning model that will determine whether or not a driver is too impaired to drive. Your training set contains 20 attributes describing driver state and behavior (swerviness, blink rate etc.) Along with a class label describing whether or not a collision occurred during the trip. Your training set consists of 2000 collisions and 2000 non-collisions. After training a random forest, your test set accuracy is 89%. Your supervisor asks whether you could get better results with a larger training set. How might you find the answer to her question?

2. **AdaBoost**

The AdaBoost algorithm can be summarized as follows:

Repeat the following for k iterations:

- Create a training set by sampling with replacement according to the sample weights $\mathbf{w}$ (initially all weights are equal.)
- Train a base C_i classifier on the sampled data.
- Apply the classifier to all training data and calculate the weighted error:

$$\epsilon_i = \sum_{j=1}^{N} w_j I\left(C_i(x_j) \neq y_j\right)$$

- Calculate the importance of the new classifier:

$$\alpha_i = \frac{1}{2} \ln \frac{1-\epsilon_i}{\epsilon_i}$$

- Update the sample weights according to:

$$w_j^{(i+1)} = \frac{w_j^{(i)}}{Z_i} \times \begin{cases} e^{-\alpha_i} & \text{if } C_i(x_j) = y_j \\ e^{\alpha_i} & \text{if } C_i(x_j) \neq y_j \end{cases}$$

where Z_i is selected to make the weights sum to 1.

Consider the following training set and initial weights:

x:	.1	.3	.6	.9
y:	-1	1	1	-1
$\mathbf{w}_0$:	.25	.25	.25	.25

(a) Assume that C_0 has been created, with the following result:

x:	.1	.3	.6	.9
correct?	N	Y	Y	Y

What is ϵ_0? What is α_0? Show your work.

(b) Fill in the table below with the updated weights. Show your work.

x:	.1	.3	.6	.9
$\mathbf{w}_1$:				

(c) Now assume that C_1 performs as follows:

x:	.1	.3	.6	.9
correct?	Y	Y	Y	N

Calculate ϵ_1, α_1 and the updated weights:

x:	.1	.3	.6	.9
$\mathbf{w}_2$:				

(d) Given that C_0 classifies a particular point as -1 and C_1 classifies it as 1, what class will be selected by the ensemble? Justify your answer.

3. **Random Forests**

(a) Bagging is an ensemble method that involves training multiple base classifiers on different subsets of the original training data, then allowing those classifiers to vote. How is the Random Forest algorithm different from just applying bagging to our standard decision tree algorithm? Based on your answer, do you think that the Random Forest improvement is likely be helpful for one-dimensional training data?

(b) Random forests generally perform better than individual decision trees. Can you imagine a situation where a simple decision tree may be preferable to a random forest?

(c) The scikit-learn implementations of both the `DecisionTreeClassifier` and the `RandomForestClassifier` classes have an attribute named `feature_importances_`. The documentation for this attribute just says "Return the feature importances (the higher, the more important the feature)." How might you calculate feature importance from a decision tree? How could your approach be extended to a random forest? How might knowing "feature importance" be useful?