1. Assume you are training a logistic regression classifier. Recall that logistic regression learns a score similar to perceptron, $\mathbf{w}^T\mathbf{x}$, which then gets plugged into the logistic function to convert the score to a probability.

a) If $\mathbf{w}^T\mathbf{x} = 0$, what is the probability that Y=1?
0.5

b) If $\mathbf{w}^T\mathbf{x} = -5.0$, which is more probable, Y=1 or Y=0?
Y=0

c) Suppose $\mathbf{w}^T\mathbf{x} = 10.0$, where x is a training instance and the training data are linearly separable. Suppose you train the classifier again, this time using a larger L2 penalty (heavier regularization). Would the score increase, decrease, or stay the same?
It would most likely decrease

2. For each hyperparameter below, answer the question: If you increase the hyperparameter, does this increase or decrease variance?

a) k in k-nearest-neighbors classification
Decrease.
With large k, you are more likely to predict the most common class in the data, rather than predicting the class that is a good fit to the particular instance.

b) λ in the logistic regression objective: $L(\mathbf{w}) + \lambda \|\mathbf{w}\|^2$
Decrease.
Higher lambda means more regularization.

c) C in the SVM objective function: $\frac{1}{2} \|\mathbf{w}\|^2 + C\, L(\mathbf{w})$
Increase.
Higher C means more sensitivity to error.

3. You try to build a classifier on a dataset. You experiment with perceptron, logistic regression, and a linear SVM, but the training accuracy is always near 50%. Assume you tuned the algorithms as best you could.

Provide a suggestion for something that could potentially improve the training accuracy, and explain why this might help.

If this happens, a likely explanation is that your data are not linearly separable, and no linear separator does a good job of distinguishing the classes. Possible remedies are to try nonlinear approaches, such as k-nearest-neighbors or a kernel SVM. Another approach would be to create new features to make the data linearly separable.

4. We learned two methods for using binary classifiers for multiclass classification: one-vs-rest and all-pairs. Suppose you develop a method that combines these two techniques for a particular problem. Suppose you have 10 classes, and you split them into two groups of 5 classes. The way an instance gets classified in your new technique is that first a classifier predicts which of the two groups the instance belongs to, then you use an all-pairs approach to classify the instance as one of the 5 classes in the group.

a) Compared to one-vs-rest, does this new technique require more or fewer classifiers? More

b) Compared to all-pairs, does this new technique require more or fewer classifiers? Fewer

c) In the traditional all-pairs approach, there are “10 choose 2” different classifiers. Write an expression for the number of classifiers in this new technique.
1 + 2 * “5 choose 2”

5. Suppose you have a feature describing eye color, and it can have one of four categorical values: “brown”, “green”, “blue”, “other”. To convert this feature to a numerical feature, you replace the four values with 1, 2, 3, 4. Explain why this is not a good way to convert this feature to numerical.

Assigning numeric values to one color variable would mean that “green” is treated as a larger value than “brown”, for example. This representation assumes there is an ordering of the colors that doesn’t exist.

The best way to change the instances would be to add 4 new features that are of binary values that mean “is Brown” or “is Green”, i.e. one-hot encoding.

6. Suppose you have a training dataset with 1000 instances (describing medical records of patients) and 3 features (blood pressure, temperature, and heart rate). Temperature has low variance, while blood pressure and heart rate both have high variance across patient records. During preprocessing, you discover that one record has an invalid heart rate value (a value of “00”, which appears to be a typo).

Describe two methods for handling this incorrect value. Then say which method would be better in this situation, and why.

One way we could handle the incorrect value would be to drop the row with the invalid heart rate. We could also fill in the empty row with the average for the entire feature. In this case, dropping the empty row is probably better because heart rate has high variance (so the average is not informative) and that instance represents only 0.1% of the dataset (so removing it will have little effect).

7. The following instances are not linearly separable. In this problem, create a new feature called x_4 that is a function of the first three features and makes the data linearly separable, meaning you could create a linear classifier that can perfectly classify the training data.

x_1	x_2	x_3	y
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1
0	0	1	0

$x_4 = x_1 * x_3$