1. Suppose you want to classify pictures as outdoor/indoor and daytime/nighttime. Should you implement two Logistic Regression classifiers or one Softmax Regression classifier? Why?

2. Implement Batch Gradient Descent with early stopping for Softmax Regression (without using Scikit-Learn)

3. Can you use logistic regression for classification between more than two classes?

4. Suppose that you are trying to predict whether a consumer will recommend a particular brand of chocolate or not. Let us say your hypothesis function outputs h(x)=0.55 where h(x) is the probability that y=1 (or that a consumer recommends the chocolate) given any input x. Does this mean that the consumer will recommend the chocolate?

5. Why can't we use the mean square error cost function used in linear regression for logistic regression?

6. In classification problems like logistic regression, classification accuracy alone is not considered a good measure. Why?

7. What is the fundamental idea behind Support Vector Machines?

8. What is a support vector?

9. Why is it important to scale the inputs when using SVMs?

10. Can an SVM classifier output a confidence score when it classifies an instance? What about a probability?

11. Should you use the primal or the dual form of the SVM problem to train a model on a training set with millions of instances and hundreds of features?

12. Say you’ve trained an SVM classifier with an RBF kernel, but it seems to underfit the training set. Should you increase or decrease γ (gamma)? What about C?

13. Train a LinearSVC on a linearly separable dataset. Then train an SVC and a SGDClassifier on the same dataset. See if you can get them to produce roughly the same model.

14. Train an SVM classifier on the MNIST dataset. Since SVM classifiers are binary classifiers, you will need to use one-versus-the-rest to classify all 10 digits. You may want to tune the hyperparameters using small validation sets to speed up the process. What accuracy can you reach?

15. Train an SVM regressor on the California housing dataset.

16. What is the approximate depth of a Decision Tree trained (without restrictions) on a training set with one million instances?

17. Is a node’s Gini impurity generally lower or greater than its parent’s? Is it generally lower/greater, or always lower/greater?

18. If a Decision Tree is overfitting the training set, is it a good idea to try decreasing max\_depth?

19. If a Decision Tree is underfitting the training set, is it a good idea to try scaling the input features?

20. If it takes one hour to train a Decision Tree on a training set containing 1 million instances, roughly how much time will it take to train another Decision Tree on a training set containing 10 million instances?

21. If your training set contains 100,000 instances, will setting presort=True speed up training?

22. Train and fine-tune a Decision Tree for the moons dataset by following these steps:

1. Use make\_moons(n\_samples=10000, noise=0.4) to generate a moons dataset.
2. Use train\_test\_split() to split the dataset into a training set and a test set.
3. Use grid search with cross-validation (with the help of the GridSearchCV class) to find good hyperparameter values for a DecisionTreeClassifier.

Hint: try various values for max\_leaf\_nodes.

1. Train it on the full training set using these hyperparameters, and measure your model’s performance on the test set. You should get roughly 85% to 87% accuracy.

23. Grow a forest by following these steps:

1. Continuing the previous exercise, generate 1,000 subsets of the training set, each containing 100 instances selected randomly. Hint: you can use ScikitLearn’s ShuffleSplit class for this.
2. Train one Decision Tree on each subset, using the best hyperparameter values found in the previous exercise. Evaluate these 1,000 Decision Trees on the test set. Since they were trained on smaller sets, these Decision Trees will likely perform worse than the first Decision Tree, achieving only about 80% accuracy.
3. Now comes the magic. For each test set instance, generate the predictions of the 1,000 Decision Trees, and keep only the most frequent prediction (you can use SciPy’s mode() function for this). This approach gives you majority-vote predictions over the test set.
4. Evaluate these predictions on the test set: you should obtain a slightly higher accuracy than your first model (about 0.5 to 1.5% higher).