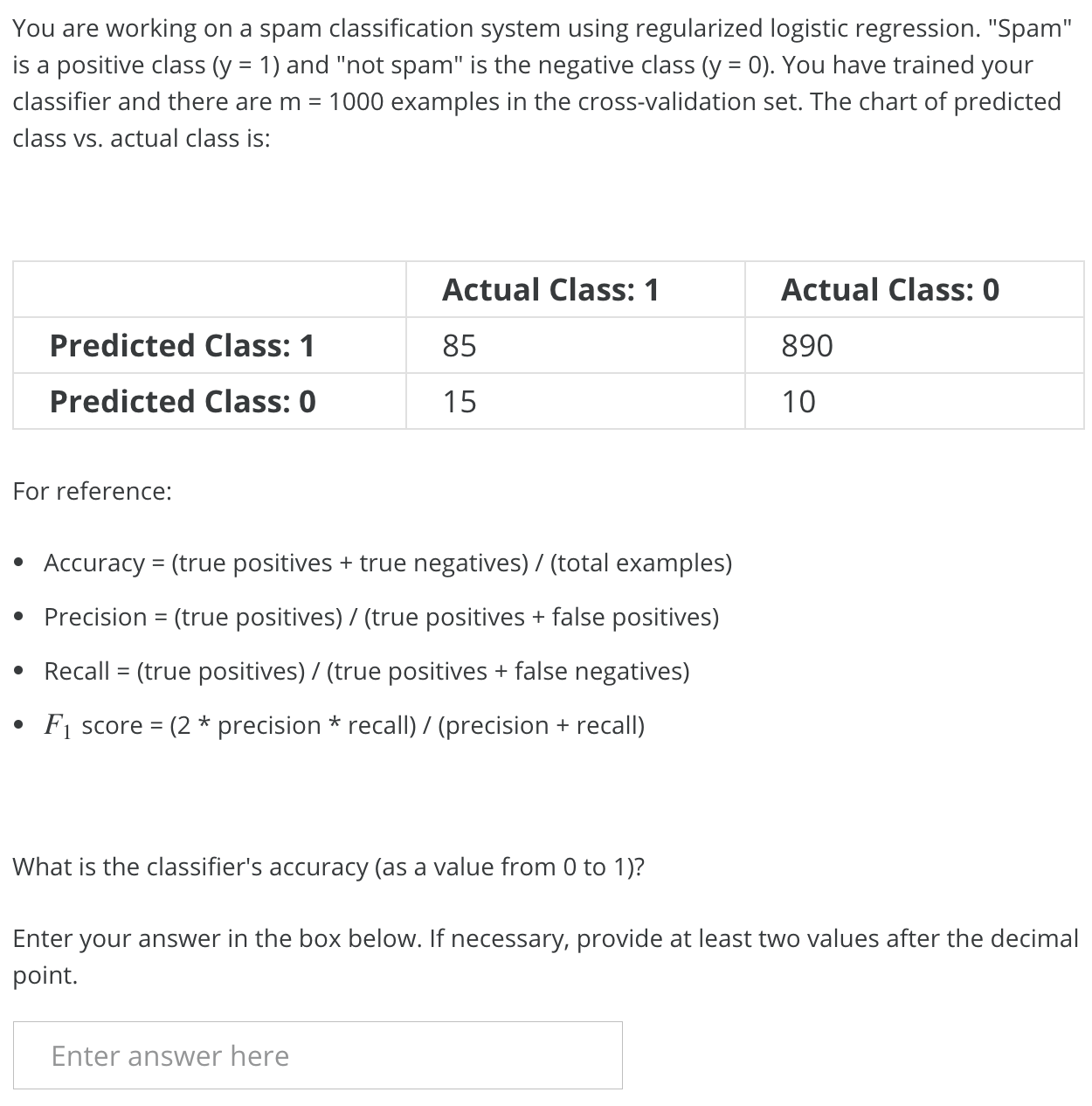
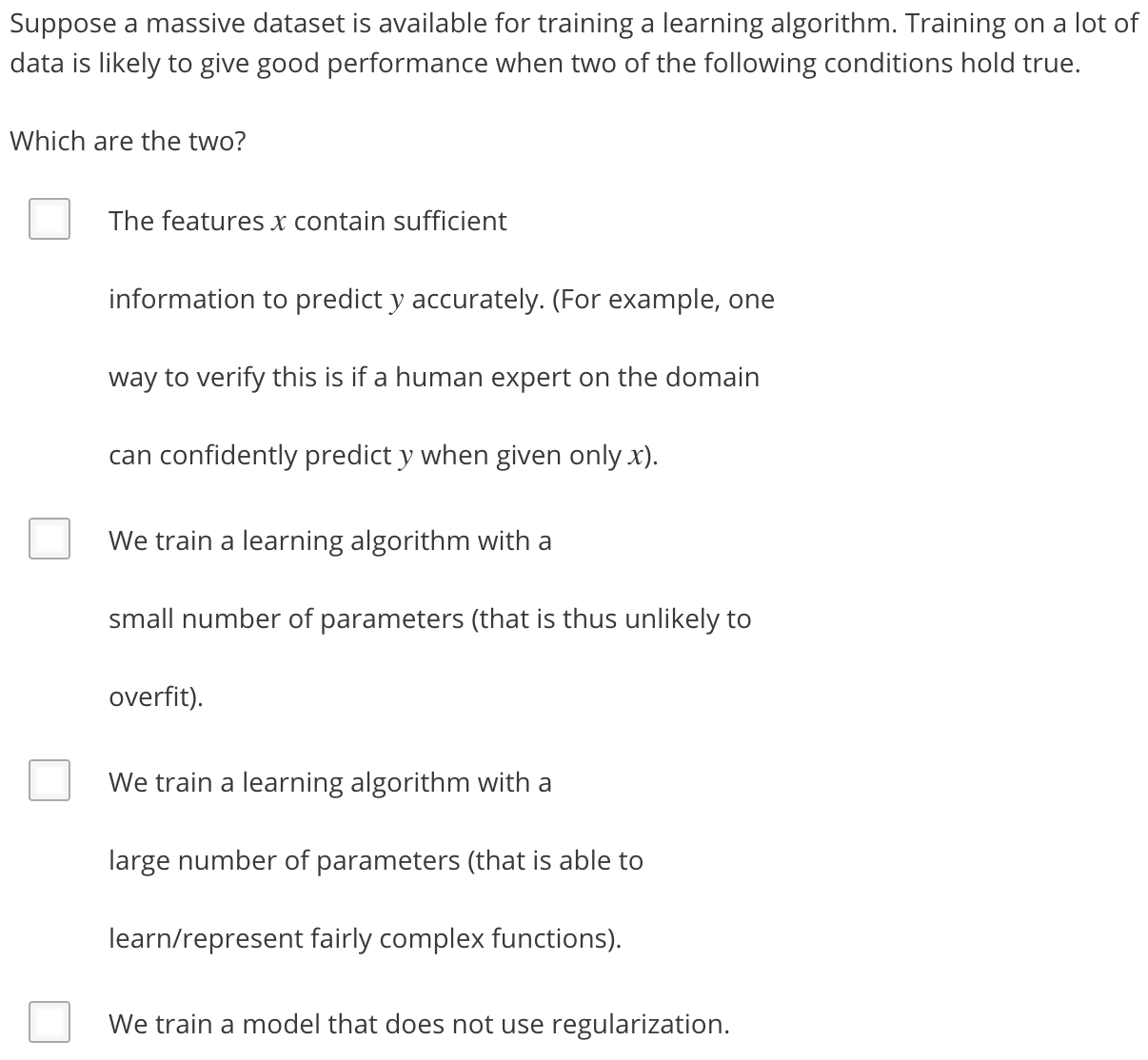
**Machine Learning Week 6 Quiz 2 (Machine Learning System Design) Stanford Coursera**

**Question 1**

[](https://github.com/mGalarnyk/datasciencecoursera/blob/master/Stanford_Machine_Learning/Week1/data/MachineLearningSystemDesignQ1.png)

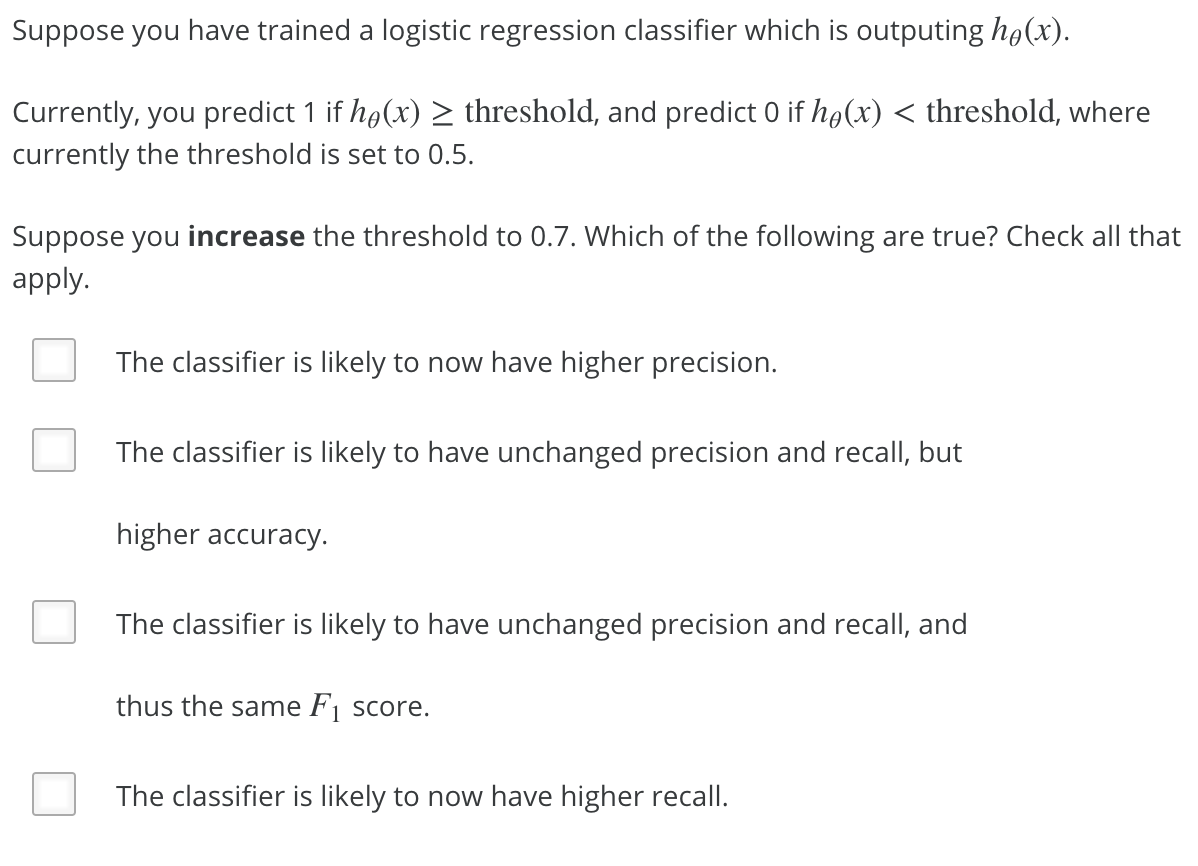
| **Statistic** | **Solution** |
| --- | --- |
| Accuracy | (85 + 10) / (1000) = .095 |
| Precision | (85) / (85 + 890) = 0.087 |
| Recall | There are 85 true positives and 15 false negatives, so recall is 85 / (85 + 15) = 0.85. |
| F1 Score | (2 \* (0.087 \* 0.85)) / (0.087 + 0.85) = 0.16 |

**Question 2**

[](https://github.com/mGalarnyk/datasciencecoursera/blob/master/Stanford_Machine_Learning/Week1/data/MachineLearningSystemDesignQ2.png)

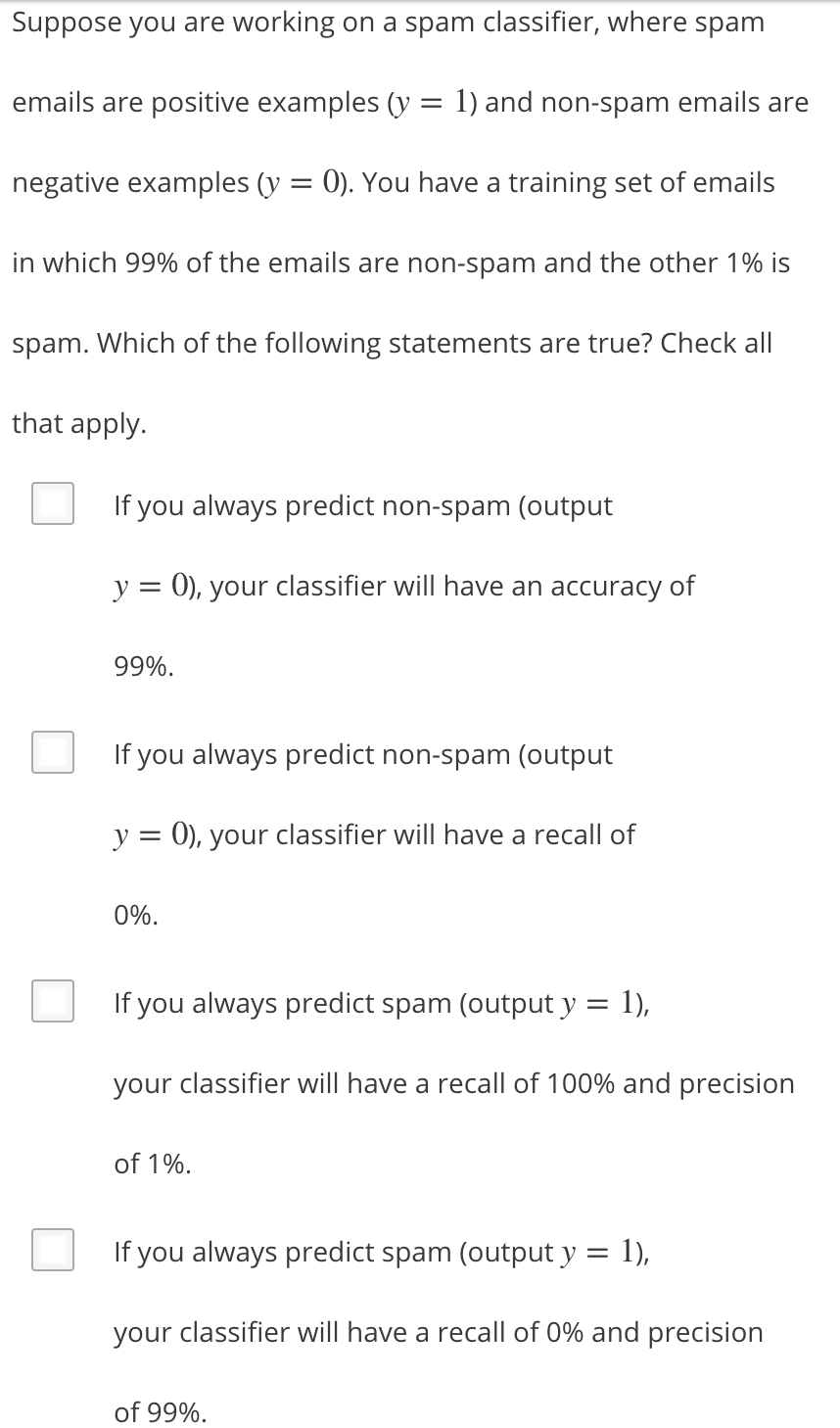
| **True/False** | **Answer** | **Explanation** |
| --- | --- | --- |
| False | We train a learning algorithm with a small number of parameters (that is thus unlikely to overfit). |  |
| True | We train a learning algorithm with a large number of parameters (that is able tolearn/represent fairly complex functions). | You should use a "low bias" algorithm with many parameters, as it will be able to make use of the large dataset provided. If the model has too few parameters, it will underfit the large training set. |
| True | The features x contain sufficient information to predict y accurately. (For example, one way to verify this is if a human expert on the domain can confidently predict y when given only x). | It is important that the features contain sufficient information, as otherwise no amount of data can solve a learning problem in which the features do not contain enough information to make an accurate prediction. |
| False | We train a model that does not use regularization. | Even with a very large dataset, some regularization is still likely to help the algorithm's performance, so you should use cross-validation to select the appropriate regularization parameter. |
| False | The classes are not too skewed. | The problem of skewed classes is unrelated to training with large datasets. |
| True | Our learning algorithm is able to represent fairly complex functions (for example, if we train a neural network or other model with a large number of parameters). | You should use a complex, "low bias" algorithm, as it will be able to make use of the large dataset provided. If the model is too simple, it will underfit the large training set. |
| False | When we are willing to include high order polynomial features of x | As we saw with neural networks, polynomial features can still be insufficient to capture the complexity of the data, especially if the features are very high-dimensional. Instead, you should use a complex model with many parameters to fit to the large training set. |
| True | A human expert on the application domain can confidently predict y when given only the features x (or more generally we have some way to be confident that x contains sufficient information to predict y accuratly) | This is a nice project commencement briefing. |

**Question 3**

[](https://github.com/mGalarnyk/datasciencecoursera/blob/master/Stanford_Machine_Learning/Week1/data/MachineLearningSystemDesignQ3.png)

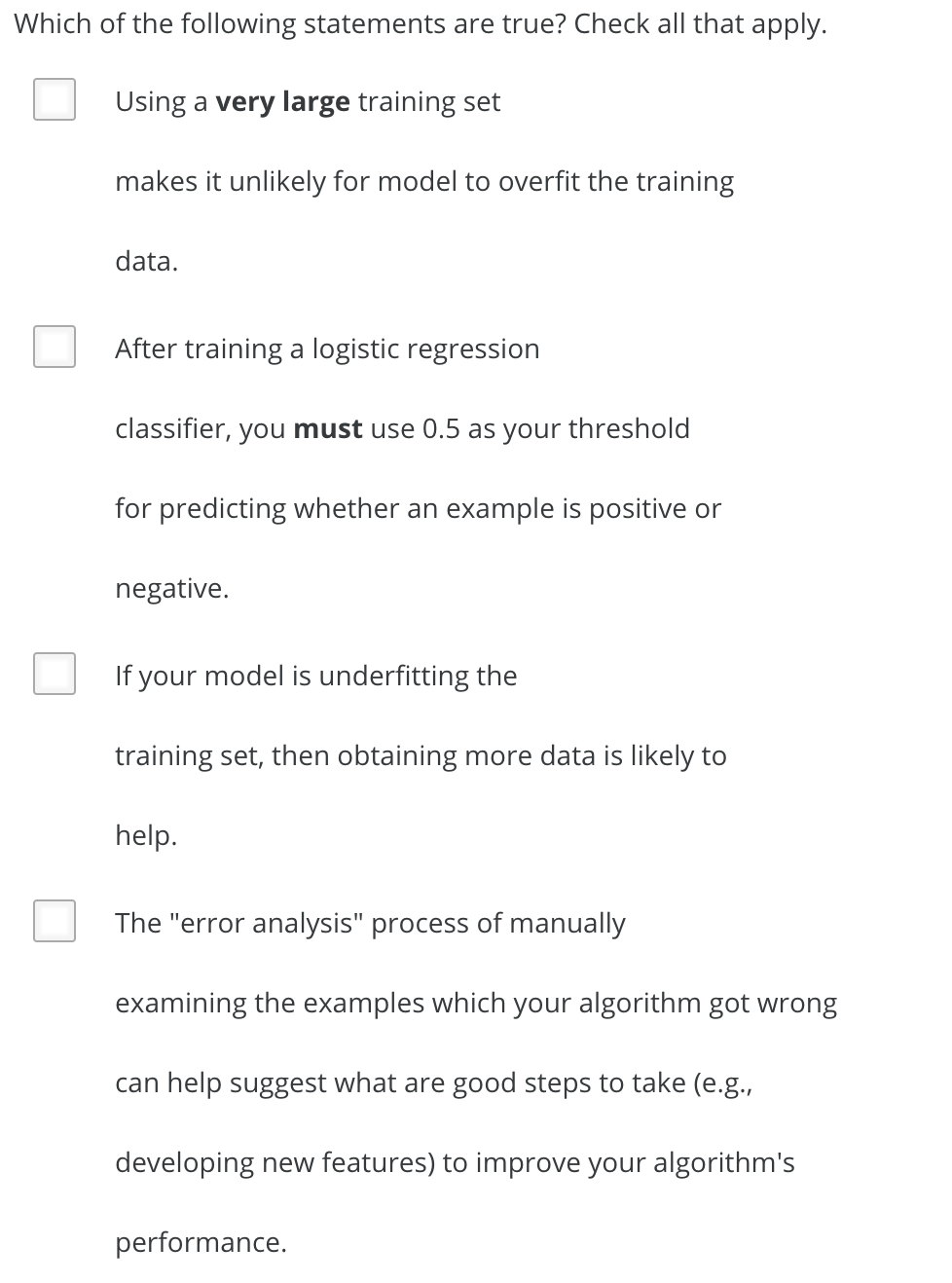
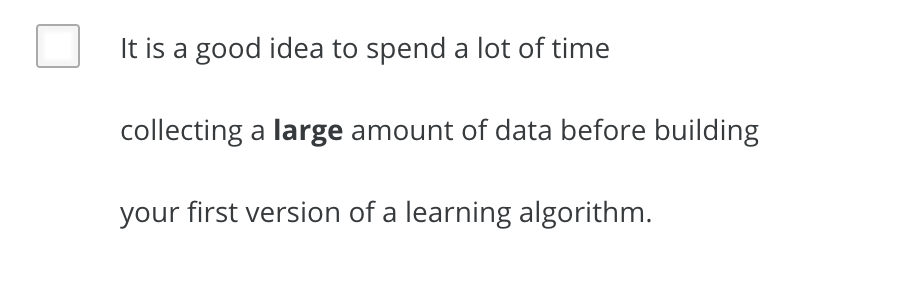
| **True/False** | **Answer** | **Explanation** |
| --- | --- | --- |
| True | The classifier is likely to now have lower recall. | Increasing the threshold means more y = 0 predictions. This increase will decrease the number of true positives and increase the number of false negatives, so recall will decrease. |
| False | The classifier is likely to have unchanged precision and recall, but lower accuracy. | By making more y = 0 predictions, we decrease true and false positives and increase true and false negatives. Thus, precision and recall will certainly change. We cannot say whether accuracy will increase or decrease. |
| False | The classifier is likely to have unchanged precision and recall, but thus the same F1 score. | By making more y = 0 predictions, we decrease true and false positives and increase true and false negatives. Thus, precision and recall will certainly change. We cannot say whether the F1 score will increase or decrease. |
| False | The classifier is likely to now have lower precision. | Increasing the threshold means more y = 0 predictions. This will decrease both true and false positives, so precision will increase, not decrease. |

**Question 4**

[](https://github.com/mGalarnyk/datasciencecoursera/blob/master/Stanford_Machine_Learning/Week1/data/MachineLearningSystemDesignQ4.png)

| **True/False** | **Answer** | **Explanation** |
| --- | --- | --- |
| True | If you always predict non-spam (output y = 0 ), your classifier will have a recall of 0%. | Since every prediction is y = 0, there will be no true positives, so recall is 0%. |
| False | If you always predict spam (output y = 1), your classifier will have a recall of 0% and precision of 99%. | Every prediction is y = 1, so recall is 100% and precision is only 1%. |
| True | If you always predict non-spam (output y = 0), your classifier will have 99% accuracy on the training set, and it will likely perform similarly on the cross validation set. | The classifier achieves 99% accuracy on the training set because of how skewed the classes are. We can expect that the cross-validation set will be skewed in the same fashion, so the classifier will have approximately the same accuracy. |
| True | If you always predict spam (output y = 1), your classifier will have a recall of 100% and precision of 1%. | Since every prediction is y = 1, there are no false negatives, so recall is 100%. Furthermore, the precision will be the fraction of examples with are positive, which is 1%. |
| False | A good classifier should have both a high precision and high recall on the cross validation set. | Different classifiers are made for different reasons with different goals. |
| True | If you always predict non-spam (output y=0), your classifier will have an accuracy of 99%. | none needed. |

**Question 5**

[](https://github.com/mGalarnyk/datasciencecoursera/blob/master/Stanford_Machine_Learning/Week1/data/MachineLearningSystemDesignQ5p1.png) [](https://github.com/mGalarnyk/datasciencecoursera/blob/master/Stanford_Machine_Learning/Week1/data/MachineLearningSystemDesignQ5p2.png)

| **True/False** | **Answer** | **Explanation** |
| --- | --- | --- |
| True | Using a **very large** training set makes it unlikely for model to overfit the training data | A sufficiently large training set will not be overfit, as the model cannot overfit some of the examples without doing poorly on the others. |
| False | It is a good idea to spend a lot of time collecting a **large** amount of data before building your first version of a learning algorithm. | You cannot know whether a huge dataset will be important until you have built a first version and find that the algorithm has high variance. |
| False | After training a logistic regression classifier, you **must** use 0.5 as your threshold for predicting whether an example is positive or negative. | You can and should adjust the threshold in logistic regression using cross validation data. |
| False | If your model is underfitting the training set, then obtaining more data is likely to help. | If the model is underfitting the training data, it has not captured the information in the examples you already have. Adding further examples will not help any more. |
| True | The "error analysis" process of manually examining the examples which your algorithm got wrong can help suggest what are good steps to take (e.g., developing new features) to improve your algorithm's performance. | This process of error analysis is crucial in developing high performance learning systems, as the space of possible improvements to your system is very large, and it gives you direction about what to work on next. |
| True | On skewed datasets (e.g., when there are more positive examples than negative examples), accuracy is not a good measure of performance and you should instead use F1 score based on the precision and recall. | This is a wonderful interview question. |
| True | The error analysis process of manually examining the examples which your algorithm got wrong can help suggest what are good steps to take (e.g. developing new features) to improve your algorithm's performance | none needed |