

## Logistic regression

### Reading

Review Murphy Sec. 8.1, 8.2, 8.3.1-8.3.2, 8.3.6. (These have been mostly covered in lecture already.)

### Problems

1. (A) Murphy Exercise 8.1, as expanded and explained below. Note that in this dataset,  $y_i = 1$  denotes spam.
  - (i) This problem may be solved with MATLAB, or with Python.
  - (ii) Murphy's statistics given for features 56 and 57 don't make sense; you can ignore them because you won't be using his statistics to solve this exercise. Instead, it's best to compute your own statistics (where needed) from the current training data.
  - (iii) For the classification error measure, use percent misclassified points, with  $\underline{w}^T \underline{x}_i \geq 0 \Rightarrow \hat{y}_i = +1$  and  $\underline{w}^T \underline{x}_i < 0 \Rightarrow \hat{y}_i = -1$  (or 0).
  - (iv) To read the csv files in Matlab we recommend using the command `readtable`. In Python, you can use libraries such as `csv` or `pandas`.
  - (v) For preprocessing method (a) note that you should compute the mean and variance for the training set and then use them to standardize the validation/test set. Matlab users are recommended to use function `zscore` to standardize the training data. You must code the standardization of the validation/test set using statistics from the train set yourself. Python users are recommended to use scikit-learn's `StandardScaler`.
  - (vi) Matlab provides the function `fitclinear`. Be sure to check its documentation – this function will run an SVM classifier if not given the proper parameters. Scikit-learn provides the `LogisticRegression` class. Pay attention to the fact that parameter `C` is **not** equal to  $\lambda$ . You may also want to change parameter `max_iter`.
  - (vii) You may code up the cross-validation loop yourself or use library functions.  
 MATLAB users may find `cvpartition` or `crossval` useful. If you choose to use `crossval`, be extra careful at the standardization step – be sure to check the documentation examples. Python users may find `StratifiedKFold` or `cross_val_score` useful. If you choose to use `cross_val_score`, be sure to use a pipeline (<https://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html>).
  - (viii) For model selection (choosing your value of  $\lambda$ ), use 5-fold cross validation, and run the cross-validation 5 times, taking average

validation errors over the multiple runs for each value of  $\lambda$ . Note that at each run you should partition the given training set randomly.

- (ix) **Report your selected value of  $\lambda$  and explain why you chose that particular  $\lambda$ .** This value of  $\lambda$  defines your “selected model”.
- (x) After choosing your value of  $\lambda$ , train again using all the training data, and then test using the test data. **Report on the following classification errors for your selected value of  $\lambda$ .** Please use a table like the example in Murphy except with 5 columns instead of 3 as follows:
  - Column 1: preprocessing method
  - Column 2: value of  $\lambda$
  - Column 3: average cross-validation error from the validation sets.
  - Column 4: error on the full given training set (trained on the full given training set)
  - Column 5: error on the full given test set (trained on the full given training set)

(B) Additional Question:

This problem pertains only to the given test data, as provided with the dataset. After using preprocessing method (c) on the given test data, use **sum of features 1-48 (total count of keywords in percentage)** as x axis, and **sum of features 49-54 (total count of special characters in percentage)** as y axis, and draw the following plots:

- (i) A scatter plot of all testing points, using different colors and marker size for spam and non-spam emails.
- (ii) For emails labeled spam, generate a 3D histogram. In Matlab, use function `hist3()`. In Python, you will need to follow and adapt this demo <https://matplotlib.org/3.1.1/gallery/mplot3d/hist3d.html>
- (iii) For emails labeled non-spam, generate a 3D histogram using the same tips given above.
- (iv) Do you notice any significant difference between the two histograms generated in (ii) and (iii)? If so, briefly describe.

2. Murphy Exercise 8.3.

For part (b), also answer the question: is Eq. (8.5) the gradient of the log likelihood, or of the negative log likelihood?

For part (c), also answer the question: **H** is positive definite implies what about the negative log likelihood function?

## Feasibility and fundamental issues of learning

### Reading

AML 1.3 (p.15) to end of Ch. 1 (p. 32). (This will be covered in Lectures 7 and 8.)

Comments on notation and terminology in AML:

- “Sample” means a set of data points or a set of marbles. We can also think of our training dataset as being a “sample”.
- $f(\underline{x})$  is the “target function”, and denotes the true function that gives the correct output (class label) for an input  $\underline{x}$ . This function is typically unknown to us. We try to find some reasonable approximation to  $f$  by learning from the training data.

### Problem

3. Suppose our “learning algorithm” uses a standard linear model for  $\hat{f}$  in a classification problem, in which there are  $D$  input variables (features), and augmented notation is used:

$$\hat{f}(\underline{x}) = \text{sgn}\left(\underline{w}^T \underline{x}\right)$$

in which  $\text{sgn}(u) \triangleq 1 \cdot \mathbb{I}[u > 0] - 1 \cdot \mathbb{I}[u < 0]$ , and  $\mathbb{I}[\cdot]$  denotes the indicator function. The learning algorithm picks the best weight vector  $\hat{\underline{w}}$  using the training data  $\mathcal{D}$ , based on minimizing some objective function  $J(\underline{w}, \mathcal{D})$ , with each component of  $\underline{w}$  restricted to:

$$w_0 = 1; \quad w_j \in \{1, 2\} \quad \forall j \in \{1, 2, \dots, D\}.$$

- (a) How many elements (hypotheses) are there in the hypothesis set  $\mathcal{H}$ ?
- (b) How would the Hoeffding Inequality be applied to this case? That is, give an expression, if possible, for an upper bound on  $P\left[\left|E_{\mathcal{D}}(\hat{h}) - E_{out}(\hat{h})\right| > \varepsilon\right]$ .