Exercise Sheet 7

Due: 16.01.2019, 09:00

Download the following files

- Train data: https://pjreddie.com/media/files/mnist_train.csv
- Test data: https://pjreddie.com/media/files/mnist_test.csv
- Jupyter notebook: LogisticRegression.ipynb from ISIS

The training and test datasets are csv-versions of the MNIST dataset: http://yann.lecun.com/exdb/mnist

Exercise 7.1 (optional, bonus)

Let $(x_1, y_1), \dots, (x_m, y_m) \in \mathbb{R}^{n+1} \times \{0,1\}$ be a training set with augmented inputs $x_i = (1, x_{i1}, \dots, x_{in})$. Consider the L_2-regularized loss function

$$J(w) = -\frac{1}{m}\sum_{i=1}^{m} y_i \log(\sigma(w^T x_i)) + (1 - y_i)\log(1 - \sigma(w^T x_i)) + \frac{\lambda}{2m}\sum_{j=1}^{n} w_j^2$$

Derive the loss function and its gradient in matrix notation.

Exercise 7.2

Show that the logistic loss and the squared error loss are both Fisher consistent.

Exercise 7.3

Implement logistic regression with L_2-regularization, optimized by gradient descent. For this, complete the fit() and predict() methods of the class LogisticRegression in the Jupyter notebook. The fit() method should compute the empirical risk (loss) and the accuracy on the training set after each update and return a list LOSS of all loss values and a list ACCURACY of all accuracy values. The method predict should return predicted labels rather than continuous values.

Exercise 7.4

The method get_toy_data() generates two linearly separable classes. Fit a logistic regression model without regularization and one with regularization and add the decision boundaries to the scatter plot. Try to minimize the training loss as much as possible. Plot the LOSS and ACCURACY curves for both models. Discuss your results.

Exercise 7.5

1. Apply logistic regression to two (arbitrary) classes of the MNIST Dataset. Use the function load_mnist() to load two classes from the MNIST dataset. Train the model on the training set, use the LOSS curve to check whether gradient descent works properly. Plot the LOSS and ACCURACY curves. Compute accuracy on the training and on the test sets. Discuss your results. Hint: Most of this is already given in the template! You only have to define the fitting parameters.
2. Find and plot one test example each where the trained classifier is (a) most sure (b) most unsure. Hint: You only have to implement the class methods most_sure(), most_unsure().
3. Plot the weights of the fitted model as image. Use the plot_mnist() function for this. Interpret the image.

Using Scikit-Learn

Scikit-Learn should only be used if indicated by the task. You can use Scikit-Learn in another task, A say, in order to be able to solve further tasks building upon task A. But then you'll not receive a point in task A.