Machine Learning
Exercise 4

Marc Toussaint
Machine Learning & Robotics lab, U Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany

May 7, 2018

1 Discriminative Function in Logistic Regression

Logistic Regression (ML 03 Part 1, slides 7 + 8) defines class probabilities as proportional to the exponential of a discriminative function:

$$P(y|x) = \frac{\exp f(x,y)}{\sum_{y'} \exp f(x,y')}$$

Prove that, in the binary classification case, you can assume $f(x,0) = 0$ without loss of generality.
This results in

$$P(y=1|x) = \frac{\exp f(x,1)}{1+\exp f(x,1)} = \sigma(f(x,1)).$$

(Hint: first assume $f(x,y) = \phi(x,y)^\top \beta$, and then define a new discriminative function f' as a function of the old one, such that $f'(x,0) = 0$ and for which $P(y|x)$ maintains the same expressibility.)

2 Logistic Regression

On the course webpage there is a data set `data2Class.txt` for a binary classification problem. Each line contains a data entry (x,y) with $x \in \mathbb{R}^2$ and $y \in \{0,1\}$.
a) Compute the optimal parameters β (perhaps also the mean neg-log-likelihood, $-\frac{1}{n}\log L(\beta)$) of logistic regression using linear features. Plot the probability $P(y=1\,|\,x)$ over a 2D grid of test points. Tips:

- Recall the objective function, and its gradient and Hessian that we derived in the last exercise:

$$\begin{aligned}
L(\beta) &= -\sum_{i=1}^n \log P(y_i\,|\,x_i) + \lambda \|\beta\|^2 \\
&= -\sum_{i=1}^n \Big[y_i \log p_i + (1-y_i)\log[1-p_i] \Big] + \lambda\|\beta\|^2 \\
\nabla L(\beta) &= \frac{\partial L(\beta)}{\partial \beta}^\top = \sum_{i=1}^n (p_i - y_i)\ \phi(x_i) + 2\lambda I \beta = X^\top(p-y) + 2\lambda I\beta \\
\nabla^2 L(\beta) &= \frac{\partial^2 L(\beta)}{\partial \beta^2} = \sum_{i=1}^n p_i(1-p_i)\ \phi(x_i)\ \phi(x_i)^\top + 2\lambda I = X^\top W X + 2\lambda I \\
&\text{where}\quad p(x) := P(y\!=\!1\,|\,x) = \sigma(\phi(x)^\top\beta)\ ,\quad p_i := p(x_i)\ ,\quad W := \mathrm{diag}(p \circ (1-p))
\end{aligned}$$

- Setting the gradient equal to zero can't be done analytically. Instead, optimal parameters can quickly be found by iterating Newton steps: For this, initialize $\beta = 0$ and iterate

  $$\beta \leftarrow \beta - (\nabla^2 L(\beta))^{-1}\ \nabla L(\beta)\ .$$

  You usually need to iterate only a few times ($\sim$10) til convergence.

- As you did for regression, plot the discriminative function $f(x) = \phi(x)^\top \beta$ or the class probability function $p(x) = \sigma(f(x))$ over a grid.

Useful gnuplot commands:

```
splot [-2:3][-2:3][-3:3.5] 'model' matrix \
   us ($1/20-2):($2/20-2):3 with lines notitle
plot [-2:3][-2:3] 'data2Class.txt' \
   us 1:2:3 with points pt 2 lc variable title 'train'
```

b) Compute and plot the same for quadratic features.