Linear Regression, Logistic Regression, and Perceptrons

Problem	Method	Model	Objective	Stochastic Gradient Descent Update Rule
Regression	Linear Regression	$h_{\vec{w}}(\vec{x}) = \vec{w} \cdot \vec{x} = \sum_j w^j x^j$	Squared Error	$w^j \leftarrow w^j - \lambda \cdot (h_{\vec{w}}(\vec{x}_i) - y_i) \cdot x_i^j$
Classification	Linear Regression	$h_{\vec{w}}(\vec{x}) = \vec{w} \cdot \vec{x}$	Squared Error	$w^j \leftarrow w^j - \lambda \cdot (h_{\vec{w}}(\vec{x}_i) - y_i) \cdot x_i^j$
Classification	Logistic Regression	$h_{\vec{w}}(\vec{x}) = 1/(1 + e^{-\vec{w} \cdot \vec{x}})$	Squared Error	$w^j \leftarrow w^j - \lambda \cdot (h_{\vec{w}}(\vec{x}_i) - y_i) \cdot h_{\vec{w}}(\vec{x}_i) \cdot (1 - h_{\vec{w}}(\vec{x}_i)) \cdot x_i^j$
Classification	Logistic Regression	$h_{\vec{w}}(\vec{x}) = 1/(1 + e^{-\vec{w} \cdot \vec{x}})$	Log Likelihood	$w^j \leftarrow w^j - \lambda \cdot (h_{\vec{w}}(\vec{x}_i) - y_i) \cdot x_i^j$
Classification	Perceptron	$h_{\vec{w}}(\vec{x}) = \text{sign}(\vec{w} \cdot \vec{x})$	Mistakes	$w^j \leftarrow w^j - \lambda \cdot (h_{\vec{w}}(\vec{x}_i) - y_i) \cdot x_i^j$

Notes:

- The labels y are typically encoded as $\{0, 1\}$ when using linear or logistic regression and $\{-1, +1\}$ when using the perceptron algorithm.
- Squared error is $\sum_i (h_{\vec{w}}(\vec{x}_i) - y_i)^2$.
- Log likelihood is $\sum_i \big(y_i \cdot \log(h_{\vec{w}}(\vec{x}_i)) + (1 - y_i) \cdot \log(1 - h_{\vec{w}}(\vec{x}_i))\big)$.
- Mistakes is simply the number of instances where $h_{\vec{w}}(\vec{x}_i) \neq y_i$.
- The perceptron training algorithm has the update rule
  $w^j \leftarrow w^j + x_i^j$
  for mistakes on positive examples and
  $w^j \leftarrow w^j - x_i^j$
  for mistakes on negative examples (if the data has not been transformed to have only positive instances). Now consider the perceptron update rule given in the table above:
  $w^j \leftarrow w^j - \lambda \cdot (h_{\vec{w}}(\vec{x}_i) - y_i) \cdot x_i^j$
  Assuming $\{-1, +1\}$ labels, we note that $h_{\vec{w}}(\vec{x}_i) - y_i$ is 0 if x_i is correctly classified; it is -2 if x_i is a misclassified positive example; and it is $+2$ if x_i is a misclassified negative example. Thus, the perceptron update rule given in the table above is equivalent to the standard perceptron update rule(s) when $\lambda = 1/2$.
- Finally, we note that the stochastic gradient descent update rule is identical for (1) linear regression with a squared error objective, (2) logistic regression with a log likelihood objective, and (3) perceptron with a mistake count objective. However, in each case, the model is different: $\vec{w} \cdot \vec{x}$, $1/(1 + e^{-\vec{w} \cdot \vec{x}})$, and $\text{sign}(\vec{w} \cdot \vec{x})$, respectively.