From Convex to Nonconvex: a Loss Function Analysis for Binary Classification

L. Zhao, M. Mammadov, J. Yearwood, 2010 IEEE International Conference on Data Mining Workshops
• Optimize the precision@k. (say, k=5)
• What linear classifier gives the good solution?
Classification and losses

- Tikhonov regularization:

\[
\min_{f \in \mathcal{H}} H[f] = \frac{1}{l} \sum_{i=1}^{l} V(f(x_i), y_i) + \lambda \|f\|_K^2
\]

- For classification, V: 0-1-loss
Other, smoother approximations

- Convex losses are chosen, (mainly?) because of computational reasons
- All these losses punish outlier objects very heavily
Some non convex losses in literature

- Phi-loss,
  \[ V(f(x), y) = \begin{cases} 
  1 - y f(x) & \text{if } 0 \leq y f(x) \leq 1 \\
  1 - \operatorname{Sign}(y f(x)) & \text{otherwise} 
\end{cases} \]

- 'sigmoid'-loss,
  \[ V(f(x), y) = \begin{cases} 
  (1.2 - \gamma) - \gamma y f(x) & \text{if } -1 \leq y f(x) \leq 0 \\
  (1.2 - \gamma) - \frac{(1.2 - 2\gamma) y f(x)}{\gamma/(1 - \theta)} & \text{if } 0 < y f(x) \leq \theta \\
  \gamma/(1 - \theta) - \frac{\lambda y f(x)}{(1-\theta)} & \text{if } \theta < y f(x) \leq 1 
\end{cases} \]

- normalized sigmoid,
  \[ V(f(x), y) = 1 - \tanh(\lambda y f(x)) \]

- 2-layer neural network loss
  \[ (1 - \frac{1}{1 + \exp(-y f(x))})^2 \]
Their proposal

• Smoothed 0-1-loss

\[ V(t_i) = \begin{cases} 
  0 & t_i > 1 \\
  \frac{1}{4} t_i^3 - \frac{3}{4} t_i + \frac{1}{2} & -1 \leq t_i \leq 1 \\
  1 & t_i < -1 
\end{cases} \]

• First order differentiable
• Use the quasisecant method to optimise
• Initialisation is crucial
• In practice: use SVM/hinge-loss to initialise
Classification results

- Optimize give loss, evaluate classification performance

### TABLE II

**Comparison of classification accuracy for both training and test sets**

<table>
<thead>
<tr>
<th></th>
<th>Smoothed 0-1 Loss Smooth, Nonconvex</th>
<th>Hinge Loss Nonsmooth, Convex</th>
<th>Square Loss Smooth, Convex</th>
<th>Ramp Loss Nonsmooth, Nonconvex</th>
<th>Normalized Sigmoid Loss Smooth, Nonconvex</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Train (%)</td>
<td>Test (%)</td>
<td>Train (%)</td>
<td>Test (%)</td>
<td>Train (%)</td>
</tr>
<tr>
<td>liver</td>
<td>73.34%</td>
<td><strong>72.62%</strong></td>
<td>71.08%</td>
<td>71.45%</td>
<td>70.60%</td>
</tr>
<tr>
<td>diabetes</td>
<td><strong>78.90%</strong></td>
<td><strong>77.98%</strong></td>
<td>77.18%</td>
<td>77.06%</td>
<td>77.33%</td>
</tr>
<tr>
<td>heart</td>
<td>85.93%</td>
<td>84.07%</td>
<td>85.76%</td>
<td>84.44%</td>
<td>85.68%</td>
</tr>
<tr>
<td>australian</td>
<td><strong>88.71%</strong></td>
<td>86.23%</td>
<td>87.21%</td>
<td>85.65%</td>
<td>87.65%</td>
</tr>
<tr>
<td>vote</td>
<td>96.76%</td>
<td><strong>94.23%</strong></td>
<td>94.46%</td>
<td><strong>94.23%</strong></td>
<td>94.97%</td>
</tr>
</tbody>
</table>

### Comparison of average training time over 20 runs (sec.)

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Convex</th>
<th>NonConvex</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Square</td>
<td>Hinge</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>liver</td>
<td>0.031</td>
<td>0.437</td>
</tr>
<tr>
<td>diabetes</td>
<td>0.047</td>
<td>2.593</td>
</tr>
<tr>
<td>heart</td>
<td>0.062</td>
<td>5.047</td>
</tr>
<tr>
<td>australian</td>
<td>0.046</td>
<td>5.187</td>
</tr>
<tr>
<td>vote</td>
<td>0.047</td>
<td>1.109</td>
</tr>
</tbody>
</table>
For my problem

• Use a combination of sigmoid and exponential loss:

Yay!