Maximizing the Margin

Pascal Fua
IC-CVLab
Logistic Regression is Better than the Perceptron

But ....
Outliers Can Cause Problems

- Logistic regression tries to minimize the error-rate at training time.
- Can result in poor classification rates at test time.

--> Must sometime accept to misclassify a few training samples.
The orthogonal distance between the decision boundary and the nearest sample is called the **margin**.
• The larger the margin, the better!
• The logistic regression does not guarantee a large one.

How do we maximize it?
Reminder: Signed Distance

\[ \mathbf{\tilde{w}} = [w_0, w_1, \ldots, w_n] \text{ with } \sum_{i=1}^{N} w_i^2 = 1 \]

Hyperplane: \( \mathbf{x} \in \mathbb{R}^N, \mathbf{\tilde{w}} \cdot \mathbf{\tilde{x}} = 0, \text{ with } \mathbf{\tilde{x}} = [1 \mid \mathbf{x}] \).

Signed distance: \( \mathbf{\tilde{w}} \cdot \mathbf{\tilde{x}}, \text{ with } \mathbf{\tilde{w}} = [w_0 \mid \mathbf{w}] \text{ and } ||\mathbf{w}|| = 1. \)
Binary Classification in N Dimensions

Hyperplane: \( x \in \mathbb{R}^N, \tilde{w} \cdot \tilde{x} = 0 \), with \( \tilde{x} = [1 | x] \).

Signed distance: \( \tilde{w} \cdot \tilde{x} \), with \( \tilde{w} = [w_0 | w] \) and \( ||w|| = 1 \).

Problem statement: Find \( \tilde{w} \) such that

- for all or most positive samples \( \tilde{w} \cdot \tilde{x} > 0 \),
- for all or most negative samples \( \tilde{w} \cdot \tilde{x} < 0 \).
Reformulating the Signed Distance Again

\[ \mathbf{h} = \mathbf{w} = [w_1, \ldots, w_n] \]

\[ \mathbf{h} = 0: \text{Point is on the decision boundary.} \]
\[ \mathbf{h} > 0: \text{Point on one side.} \]
\[ \mathbf{h} < 0: \text{Point on the other side.} \]

\[ \mathbf{\tilde{x}} = [1, x_1, \ldots, x_N] \]

\[ \mathbf{\tilde{w}} = [w_0, w_1, \ldots, w_n] \text{ with } \sum_{i=1}^{N} w_i^2 = 1 \]

Hyperplane: \( \mathbf{x} \in \mathbb{R}^N, \mathbf{\tilde{w}} \cdot \mathbf{\tilde{x}} = 0 \), with \( \mathbf{\tilde{x}} = [1 \mid \mathbf{x}] \).

Signed distance: \( \mathbf{\tilde{w}} \cdot \mathbf{\tilde{x}} \), with \( \mathbf{\tilde{w}} = [1 \mid \mathbf{w}] \) and \( ||\mathbf{w}|| = 1 \).
Reformulated Signed Distance

$h=0$: Point is on the decision boundary.
$h>0$: Point on one side.
$h<0$: Point on the other side.

$\tilde{x} = [1, x_1, \ldots, x_N]$

$w = [w_1, \ldots, w_n]$

$\tilde{w} = [w_0 \mid w] \in \mathbb{R}^{N+1}$

$\tilde{w}' = \frac{\tilde{w}}{||w||} = \left[ \frac{w_0}{||w||} \mid \frac{w}{||w||} \right]$

Hyperplane: $x \in \mathbb{R}^N$, $\tilde{w} \cdot \tilde{x} = 0$, with $\tilde{x} = [1 \mid x]$.

Signed distance: $\tilde{w}' \cdot \tilde{x} = \frac{\tilde{w} \cdot \tilde{x}}{||w||}$, $\forall \tilde{w} \in \mathbb{R}^{N+1}$. 
Geometric Interpretation

We are going to use this to find a classifier whose decision boundary is as far as possible from all the points.

\[
\tilde{W} \cdot \tilde{x} = \frac{y(x)}{||w||}
\]

Bishop, Chapter 4.1.1
Maximum Margin Classifier

• Given a training set \( \{(x_n, t_n)_{1 \leq n \leq N}\} \) with \( t_n \in \{-1, 1\} \) and solution such that all the points are correctly classified, we have

\[
\forall n, \quad t_n(\tilde{w}_n \cdot \tilde{x}_n) = 0 .
\]

• We can write the unsigned distance to the decision boundary as

\[
d_n = t_n \frac{(\tilde{w} \cdot \tilde{x}_n)}{||\tilde{w}||}
\]

\( \tilde{w}^* = \arg\max_\tilde{w} \min_n \left( \frac{t_n \cdot (\tilde{w} \cdot x_n)}{||\tilde{w}||} \right) \)

—> A maximum margin classifier aims to maximize this distance for the point closest to the boundary, that is, maximize the minimum such distance.
Maximum Margin Classifier

\[ \hat{\mathbf{w}}^* = \arg \max_{\mathbf{w}} \min_n \left( \frac{t_n \cdot (\hat{\mathbf{w}} \cdot \mathbf{x}_n)}{\|\mathbf{w}\|} \right) \]

• Unfortunately, this is a difficult optimization problem to solve.
• We will convert it into an equivalent, but easier to solve, problem.
Maximum Margin Classifier

• The signed distance is invariant to a scaling of $\tilde{w}$:

$$\tilde{w} \rightarrow \lambda \tilde{w} : d_n = t_n \frac{(\lambda \tilde{w} \cdot \tilde{x}_n)}{||\lambda \tilde{w}||} = \frac{(\tilde{w} \cdot \tilde{x}_n)}{||\tilde{w}||}.$$

• We can choose $\lambda$ so that for the point $m$ closest to the boundary, we have

$$t_m \cdot (\tilde{w} \cdot x_m) = 1.$$

• For all points we therefore have

$$t_n \cdot (\tilde{w} \cdot x_n) \geq 1,$$

and the equality holds for at least one point.
Linear Support Vector Machine

$$\forall n, \quad t_n(\tilde{w} \cdot x_n) \geq 1$$

$$\exists n \quad t_n(\tilde{w} \cdot x_n) = 1$$

$$\Rightarrow \min_n d_n = \min_n t_n(\tilde{w} \cdot x_n) = \frac{1}{||w||}$$

- To maximize the margin, we only need to maximize $1/||w||$.

- This is equivalent to minimizing $\frac{1}{2} ||w||^2$.

- We can find max margin classifier as

  $$w^* = \min_w \frac{1}{2} ||w||^2 \text{ subject to } \forall n, \quad t_n \cdot (\tilde{w} \cdot x_n) \geq 1$$

- This is a quadratic program, which is a **convex** problem.

$$\rightarrow$$ It can be solved to optimality.
LR vs Linear SVM

- The LR decision boundary can come close to some of the training examples.
- The SVM tries to prevent that.
From Perceptron and LR to Linear SVM

Are we done yet?

No!

Logistic Regression

Perceptron

Linear SVM
Maximum Margin Classifier

• Given a training set \(\{(x_n, t_n) \leq n \leq N\}\) with \(t_n \in \{-1, 1\}\) and solution such that all the points are correctly classified, we have

\[
\forall n, \quad t_n (\tilde{w} \cdot \tilde{x}_n) \geq 1.
\]

• We can write the unsigned distance to the decision boundary as

\[
d_n = t_n \frac{(\tilde{w} \cdot \tilde{x}_n)}{||w||}
\]

\(\rightarrow\) A maximum margin classifier aims to maximize this distance for the point closest to the boundary, that is, maximize the minimum such distance.

\[
\tilde{w}^* = \arg \max \tilde{w} \min_n \left( \frac{t_n \cdot (\tilde{w} \cdot x_n)}{||w||} \right)
\]
Overlapping Classes

The data rarely looks like this. It generally looks like that.

--> Must account for the fact that not all training samples can be correctly classified!
Relaxing the Constraints

• The original problem

\[
\mathbf{w}^* = \min_{\mathbf{w}} \frac{1}{2} \| \mathbf{w} \|^2 \text{ subject to } \forall n, \ t_n \cdot (\tilde{\mathbf{w}} \cdot \mathbf{x}_n) \geq 1,
\]

cannot be satisfied.

• We must allow some of the constraints to be violated, but as few as possible.
Slack Variables

• We introduce an additional slack variable $\xi_n$ for each sample.
• We rewrite the constraints as $t_n \cdot (\mathbf{\tilde{w}} \cdot \mathbf{x}_n) \geq 1 - \xi_n$.
• $\xi_i \geq 0$ weakens the original constraints.

- If $0 < \xi_n \leq 1$, sample $n$ lies inside the margin, but is still correctly classified
- If $\xi_n \geq 1$, then sample $i$ is misclassified
Naive Formulation

\[ w^* = \min_w \frac{1}{2} ||w||^2 \]

subject to \( \forall n, \quad t_n \cdot (\tilde{w} \cdot x_n) \geq 1 - \xi_n \) and \( \xi_n \geq 0 \)

• This would simply allow the model to violate all the original constraints at no cost.

• This would result in a useless classifier.
Improved Formulation

\[ w^* = \min_{(w, \{\xi_n\})} \frac{1}{2} \|w\|^2 + C \sum_{n=1}^{N} \xi_n, \]

subject to \( \forall n, \quad t_n \cdot (\tilde{w} \cdot x_n) \geq 1 - \xi_n \) and \( \xi_n \geq 0. \)

- \( C \) is constant that controls how costly constraint violations are.
- The problem is still convex.

http://www.cristiandima.com/basics-of-support-vector-machines/
Choosing the C Parameter

C=1:
- Large margin.
- Many training samples misclassified.

C=100:
- Small margin.
- Few training samples misclassified.

Which is best?
- It depends.
- Must use cross-validation, as we did for k-Means.
Optimal vs Best

- The points can be linearly separated but the margin is still very small.
- At test time the two green circles will be misclassified.
- The margin is much larger but one training example is misclassified.
- At test time the two green circles will be classified correctly.

--> Tradeoff between the number of mistakes on the training data and the margin.
### Support Vector Machines

<table>
<thead>
<tr>
<th>Model</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic Regression</td>
<td>63.5%</td>
</tr>
<tr>
<td>Decision Trees</td>
<td>49.9%</td>
</tr>
<tr>
<td>Random Forests</td>
<td>46.3%</td>
</tr>
<tr>
<td>Neural Networks</td>
<td>37.6%</td>
</tr>
<tr>
<td>Bayesian Techniques</td>
<td>30.6%</td>
</tr>
<tr>
<td>Ensemble Methods</td>
<td>28.5%</td>
</tr>
<tr>
<td>SVMs</td>
<td>26.7%</td>
</tr>
<tr>
<td>Gradient Boosted Machines</td>
<td>23.9%</td>
</tr>
<tr>
<td>CNNs</td>
<td>18.9%</td>
</tr>
<tr>
<td>RNNs</td>
<td>12.3%</td>
</tr>
<tr>
<td>Other</td>
<td>8.3%</td>
</tr>
<tr>
<td>Evolutionary Approaches</td>
<td>5.5%</td>
</tr>
<tr>
<td>HMMs</td>
<td>5.4%</td>
</tr>
<tr>
<td>Markov Logic Networks</td>
<td>4.9%</td>
</tr>
<tr>
<td>GANs</td>
<td>2.8%</td>
</tr>
</tbody>
</table>