Object Detection

Ali Farhadi
CSE 576
We have talked about

• Nearest Neighbor
• Naïve Bayes
• Logistic Regression
• Boosting

• We saw face detection
Support Vector Machines
Linear classifiers – Which line is better?

\[ \mathbf{w} = \sum_j w^{(j)} x^{(j)} \]

Data

\[
\begin{align*}
\langle x^{(1)}_1, \ldots, x^{(m)}_1, y_1 \rangle \\
\vdots \\
\langle x^{(1)}_n, \ldots, x^{(m)}_n, y_n \rangle
\end{align*}
\]

Example i

\[
\langle x^{(1)}_i, \ldots, x^{(m)}_i \rangle \quad \text{—} \quad m \text{ features}
\]

\[ y_i \in \{-1, +1\} \quad \text{—} \quad \text{class} \]
Pick the one with the largest margin!

**Margin**: measures height of $w.x+b$ plane at each point, increases with distance

$$\gamma_j = (w.x_j + b)y_j$$

**Max Margin**: two equivalent forms

1. $$\max_{w,b} \min_j \gamma_j$$
2. $$\max_{\gamma,w,b} \gamma \quad \forall j \ (w.x_j + b)y_j > \gamma$$

$$w.x = \sum_j w^{(j)} x^{(j)}$$
How many possible solutions?

\[
\max_{\gamma, w, b} \gamma \\
\forall j \ (w \cdot x_j + b) y_j > \gamma
\]

Any other ways of writing the same dividing line?

- \( w \cdot x + b = 0 \)
- \( 2w \cdot x + 2b = 0 \)
- \( 1000w \cdot x + 1000b = 0 \)
- ....
- Any constant scaling has the same intersection with \( z=0 \) plane, so same dividing line!

Do we really want to \( \max_{\gamma, w, b} \)?
**Review: Normal to a plane**

\[ w \cdot x + b = 0 \]

**Key Terms**

- \( \bar{x}_j \) -- projection of \( x_j \) onto \( w \)
- \( \frac{w}{\|w\|} \) -- unit vector normal to \( w \)
**Idea: constrained margin**

\[ x_j = x'_j + \lambda \frac{w}{||w||} \]

**Generally:**

\[ x^+ = x^- + 2\gamma \frac{w}{||w||} \]

**Assume:** \( x^+ \) on positive line, \( x^- \) on negative

\[
\begin{align*}
    w.x^+ + b &= 1 \\
    w.\left(x^- + 2\gamma \frac{w}{||w||}\right) + b &= 1 \\
    w.x^- + b + 2\gamma \frac{w.w}{||w||} &= 1 \\
    \gamma \frac{w.w}{||w||} &= 1
\end{align*}
\]

\[
\gamma = \frac{||w||}{w.w} = \frac{1}{\sqrt{w.w}}
\]

**Final result:** can maximize constrained margin by minimizing \( ||w||_2 \)!!!
Max margin using canonical hyperplanes

\begin{align*}
\text{maximize}_{\gamma,w,b} & \quad \gamma \\
\left( w \cdot x_j + b \right) y_j & \geq \gamma, \quad \forall j \in \text{Dataset}
\end{align*}

\[ \gamma = \frac{1}{\sqrt{w \cdot w}} \]

\begin{align*}
\text{minimize}_{w,b} & \quad w \cdot w \\
\left( w \cdot x_j + b \right) y_j & \geq 1, \quad \forall j \in \text{Dataset}
\end{align*}

The assumption of canonical hyperplanes (at +1 and -1) changes the objective and the constraints!
Support vector machines (SVMs)

- Solve efficiently by quadratic programming (QP)
  - Well-studied solution algorithms
  - Not simple gradient ascent, but close
- Hyperplane defined by support vectors
  - Could use them as a lower-dimension basis to write down line, although we haven’t seen how yet
  - More on this later

Non-support Vectors:
- everything else
- moving them will not change \( w \)

Support Vectors:
- data points on the canonical lines

\[
\begin{align*}
w \cdot x + b &= +1 \\
w \cdot x + b &= 0 \\
w \cdot x + b &= -1
\end{align*}
\]

\[
\text{margin } 2\gamma
\]

\[
\min_{w,b} \quad w \cdot w
\]

\[
(w \cdot x_j + b) y_j \geq 1, \ \forall j
\]
What if the data is not linearly separable?

\[ \langle x_i^{(1)}, \ldots, x_i^{(m)} \rangle \quad \text{— m features} \]

\[ y_i \in \{-1, +1\} \quad \text{— class} \]

Add More Features!!!

\[ \phi(x) = \begin{pmatrix} 
 x^{(1)} \\
 \ldots \\
 x^{(n)} \\
 x^{(1)}x^{(2)} \\
 x^{(1)}x^{(3)} \\
 \ldots \\
 e^{x^{(1)}} \\
 \ldots 
\end{pmatrix} \]

What about overfitting?
What if the data is still not linearly separable?

\[
\text{minimize}_{w,b} \quad w \cdot w + C \#(\text{mistakes})
\]
\[
(w \cdot x_j + b) y_j \geq 1, \forall j
\]

• First Idea: Jointly minimize \( w \cdot w \) and number of training mistakes
  – How to tradeoff two criteria?
  – Pick \( C \) on development / cross validation

• Tradeoff \#(mistakes) and \( w \cdot w \)
  – 0/1 loss
  – Not QP anymore
  – Also doesn’t distinguish near misses and really bad mistakes
  – NP hard to find optimal solution!!!
Slack variables – Hinge loss

For each data point:

• If margin ≥ 1, don’t care
• If margin < 1, pay linear penalty

\[ \text{minimize}_{w, b} \quad w \cdot w + C \sum_j \xi_j \]
\[ (w \cdot x_j + b) y_j \geq 1 - \xi_j \quad \forall j \quad \xi_j \geq 0 \]

Slack Penalty \( C > 0 \):

• \( C=\infty \) → have to separate the data!
• \( C=0 \) → ignore data entirely!
• Select on dev. set, etc.
Side Note: Different Losses

Logistic regression:
\[ \sum_{i=1}^{m} \ln(1 + \exp(-y_if(x_i))) \]

Boosting:
\[ \frac{1}{m} \sum_{i} \exp(-y_if(x_i)) = \prod_{t} Z_t \]

SVM:
\[
\text{minimize}_{w,b} \quad w.w + C \sum_j \xi_j \\
(\mathbf{w}.x_j + b) y_j \geq 1 - \xi_j, \quad \forall j \\
\xi_j \geq 0, \quad \forall j
\]

Hinge loss:
\[ \xi_j = (1 - f(x_i)y_i) + \]

0-1 Loss:
\[ \delta(H(x_i) \neq y_i) \]

All our new losses approximate 0/1 loss!
What about multiple classes?
One against All

Learn 3 classifiers:
• $+$ vs $\{0,-\}$, weights $w_+$
• $-$ vs $\{0,+,+\}$, weights $w_-$
• $0$ vs $\{+,+-\}$, weights $w_0$

Output for $x$:
\[ y = \arg\max_i w_i.x \]

Any other way?

Any problems?
Learn 1 classifier: Multiclass SVM

Simultaneously learn 3 sets of weights:

• How do we guarantee the correct labels?
• Need new constraints!

For $j$ possible classes:

$$w(y_j) \cdot x_j + b(y_j) \geq w(y') \cdot x_j + b(y') + 1, \quad \forall y' \neq y_j, \quad \forall j$$
Learn 1 classifier: Multiclass SVM

Also, can introduce slack variables, as before:

\[
\minimize_{w,b} \quad \sum_y w(y).w(y) + C \sum_j \xi_j \\
\quad w(y_j).x_j + b(y_j) \geq w(y').x_j + b(y') + 1 - \xi_j, \quad \forall y' \neq y_j, \forall j \\
\quad \xi_j \geq 0, \quad \forall j
\]
What if the data is not linearly separable?

\[
\left\{ x_i^{(1)}, \ldots, x_i^{(m)} \right\} \quad \text{— } m \text{ features}
\]

\[
y_i \in \{-1, +1\} \quad \text{— class}
\]

Add More Features!!!

\[
\phi(x) = \begin{pmatrix}
x^{(1)} \\
\cdots \\
x^{(n)} \\
x^{(1)}x^{(2)} \\
x^{(1)}x^{(3)} \\
\cdots \\
e^{x^{(1)}} \\
\cdots
\end{pmatrix}
\]
SVM with a polynomial Kernel visualization

Created by: Udi Aharoni
## Comparison

<table>
<thead>
<tr>
<th>Learning Objective</th>
<th>Training</th>
<th>Inference</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Naïve Bayes</strong></td>
<td>$\max \sum_i \left[ \sum_j \log P(x_{ij}</td>
<td>y_i; \theta_j) + \log P(y_i; \theta_0) \right]$</td>
</tr>
<tr>
<td><strong>Logistic Regression</strong></td>
<td>$\max \sum_i \left[ \log(P(y_i</td>
<td>x, \theta)) + \lambda |\theta|^2 \right]$ where $P(y_i</td>
</tr>
<tr>
<td><strong>Linear SVM</strong></td>
<td>$\min \lambda \sum_i \xi_i + \frac{1}{2} |\theta|^2$ such that $y_i \theta^T x \geq 1 - \xi_i \ \forall i, \xi_i \geq 0$</td>
<td>Quadratic programming or subgradient opt.</td>
</tr>
<tr>
<td><strong>Kernelized SVM</strong></td>
<td>complicated to write</td>
<td>Quadratic programming</td>
</tr>
<tr>
<td><strong>Nearest Neighbor</strong></td>
<td>$\text{most similar features} \rightarrow \text{same label}$</td>
<td>Record data</td>
</tr>
</tbody>
</table>
Image Categorization

Training

Training Images → Image Features → Classifier Training → Trained Classifier

Training Labels

Testing

Test Image → Image Features → Trained Classifier → Prediction: Outdoor
Example: Dalal-Triggs pedestrian

1. Extract fixed-sized (64x128 pixel) window at each position and scale
2. Compute HOG (histogram of gradient) features within each window
3. Score the window with a linear SVM classifier
4. Perform non-maxima suppression to remove overlapping detections with lower scores

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05
• Tested with
  - RGB
  - LAB
  - Grayscale

Slightly better performance vs. grayscale
Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05
Histogram of gradient orientations

Orientation: 9 bins (for unsigned angles)

- Votes weighted by magnitude
- Bilinear interpolation between cells

Navneet Dalal and Bill Triggs, Histograms of Oriented Gradients for Human Detection, CVPR05
Normalize with respect to surrounding cells

\[ L_2 \text{- norm} : v \mapsto \frac{v}{\sqrt{||v||^2_2 + \epsilon^2}} \]
# features = 15 x 7 x 9 x 4 = 3780

# orientations

# cells

# normalizations by neighboring cells
Training set
0.16 = w^T x - b

\text{sign}(0.16) = 1

\implies \text{pedestrian}
Detection examples
Each window is separately classified