Machine Learning Methods
Playing with Kernels

Zdeněk Žabokrtský, Ondřej Bojar
Institute of Formal and Applied Linguistics
Faculty of Mathematics and Physics
Charles University, Prague

Tue Apr 10, 2018
Outline

- Regularization parameter C in SVM.
- Linear Kernel: $k(x, y) = x \cdot y$
- Kernel Trick.
- Polynomial Kernel: $k(x, y) = (\gamma \ast x \cdot y + \text{coeff0})^{\text{degree}}$
- RBF Kernel: $k(x, y) = \exp(-\gamma \|x - y\|^2); \gamma > 0$
  ... including their parameters
- Cross-validation Heatmap
- Multi-class SVM
  - For the PAMAP-easy dataset.
  - Regularization parameters.
  - Inseparable classes.

Regularization (C) in linear SVM

\[ k(x, y) = x \cdot y \]

(Linear kernel = no kernel)

The parameter \( C \) in (linear) SVM:

- sets the weight of the sum of slack variables.
- serves as a regularization parameter.
- controls the number of support vectors.

<table>
<thead>
<tr>
<th>( C )</th>
<th>Penalty for Errors</th>
<th>Number of points considered</th>
<th>Margin</th>
<th>Bias</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Low</td>
<td>Many</td>
<td>Wide</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>High</td>
<td>High</td>
<td>Few</td>
<td>Narrow</td>
<td>Low</td>
<td>High</td>
</tr>
</tbody>
</table>

Think \( C \) for Variance.
SVM Linear $C=0.1$
SVM Linear $C=0.2$
SVM Linear $C=0.5$
SVM Linear $C=1$
SVM Linear $C=5$
SVM Linear $C=10$
SVM Linear $C=20$
SVM Linear $C=50$
SVM Linear $C=100$
Kernel Trick

- Classifiers generally do **linear** separation.
- It can be very difficult to come up with features that allow for linear separation.

“Kernel trick”: map the coordinates to another space where separation is possible:
Kernel Function to a Higher Dimension

\[ k(x, y) = xy + x^2y^2 \]  \hspace{1cm} (1)
Deep NNs: Kernel Trick on Steroids

Neurons represent information as populations of visually-evoked “features”

“Joe’s” identity manifold

“Joe”

Slides from DiCarlo (2013).
Deep NNs: Kernel Trick on Steroids

The computational crux of object and face recognition

A “good” set of visual features

== “Explicit” representation of object shape

We assume: “shape” maps to “identity” and “category”

“Joe”

Neural population

individual 2 (“Joe”)

Should be able to find it with low* number of training examples

“not Joe”

“not Joe”

individual 1 (“Sam”)

linear classifier

downstream neuron(s)

DiCarlo and Cox, TICS (2007)

Slides from DiCarlo (2013).
Deep NNs: Kernel Trick on Steroids

Slides from DiCarlo (2013).
Deep NNs: Kernel Trick on Steroids

Tangled, implicit object information

Transformation

pixel RGC LGN V1 V2 V4 IT

Slides from DiCarlo (2013).
Deep NNs: Kernel Trick on Steroids
Polynomial Kernel

\[ k(x, y) = (\gamma \ast x \cdot y + \text{coeff0})^{\text{degree}} \]
SVM Poly (degree 1)
SVM Poly (degree 2)
SVM Poly (degree 3)
SVM Poly (degree 4)
SVM Poly (degree 5)
SVM Poly (degree 6)
SVM Poly (degree 7)
SVM Poly (degree 8)
SVM Poly (degree 9)
SVM Poly (degree 3, gamma 0.05)
SVM Poly (degree 3, gamma 0.1)
SVM Poly (degree 3, gamma 0.2)
SVM Poly (degree 3, gamma 0.5)
SVM Poly (degree 3, gamma 0.7)
SVM Poly (degree 3, gamma 1)
SVM Poly (degree 3, gamma 2)
SVM Poly \( (d=3, \ g=0.5, \ coef=-2.0) \)
SVM Poly ($d=3, \ g=0.5, \ coef=-1.0$)
SVM Poly \( (d=3, \ g=0.5, \ \text{coef}=-0.50) \)
SVM Poly \((d=3, \ g=0.5, \ coef=0)\)
SVM Poly (d=3, g=0.5, coef=0.5)
SVM Poly \((d=3, \ g=0.5, \ \text{coef}=1)\)
SVM Poly (d=3, g=0.5, coef=2)
RBF Kernels

\[ k(x, y) = \exp(-\gamma \|x - y\|^2); \gamma > 0 \]

- Each training point creates its bell.
- Overall shape is the sum of the bells.
- Kind of “all nearest neighbours”.

... Totally “flips” the space:
- The axes are now closeness to other objects.
# RBF Kernel Parameters

<table>
<thead>
<tr>
<th>$C$</th>
<th>Decision Surface</th>
<th>Model</th>
<th>Bias</th>
<th>Variance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>Smooth</td>
<td>Simple</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>High</td>
<td>Peaked</td>
<td>Complex</td>
<td>Low</td>
<td>High</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>gamma</th>
<th>Affected Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low</td>
<td>can be far from training examples</td>
</tr>
<tr>
<td>High</td>
<td>must be close to training examples</td>
</tr>
</tbody>
</table>

- Does higher gamma lead to higher variance?
- Choice critical for SVM performance.
- Advised to use GridSearchCV for $C$ and gamma:
  - exponentially spaced probes
  - wide range
SVM RBF (C=0.05, gamma=2)
SVM RBF \( (C=0.1, \text{ gamma}=2) \)
SVM RBF (C=0.2, gamma=2)
SVM RBF (C=0.5, gamma=2)
SVM RBF ($C=0.6$, $\text{gamma}=2$)
SVM RBF (C=0.7, gamma=2)
SVM RBF (C=1, gamma=2)
SVM RBF (C=2, gamma=2)
SVM RBF (C=1, gamma=2)
SVM RBF (C=0.5, gamma=2)
SVM RBF (C=0.5, gamma=5)
SVM RBF (C=0.5, gamma=10)
SVM RBF \((C=0.5, \text{ gamma}=5)\)
SVM RBF \((C=0.5, \text{ gamma}=2)\)
SVM RBF (C=0.5, gamma=1)
SVM RBF (C=0.5, gamma=0.7)
SVM RBF (C=0.5, gamma=0.5)
SVM RBF (C=0.5, gamma=0.2)
SVM RBF (C=0.5, gamma=0.1)
SVM RBF (C=0.5, gamma=0.05)
Cross-validation Heatmap

Multi-class SVM

Two implementations in scikit-learn:

- **SVC**: one-against-one
  - \( n(n - 1)/2 \) classifiers constructed
  - supports various kernels, incl. custom ones

- **LinearSVC**: one-vs-the-rest
  - \( n \) classifiers trained
Default View (every 200)

- SVC with linear kernel
- SVC with RBF kernel (gamma 0.7)
- SVC with polynomial (degree 3) kernel
- LinearSVC (linear kernel)

regularization: C=1.0
Default View (every 300)

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: C=1.0
Default View (every 400)

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: C=1.0
Regularization $C=0.5$

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: $C=0.5$
Regularization $C=1$

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: $C=1.0$
Regularization $C=5$

- **SVC with linear kernel**
- **SVC with RBF kernel (gamma 0.7)**
- **SVC with polynomial (degree 3) kernel**
- **LinearSVC (linear kernel)**

regularization: $C=5.0$
Regularization $C=10$

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: $C=10.0$
Regularization $C=20$

- SVC with linear kernel
- SVC with RBF kernel (gamma 0.7)
- SVC with polynomial (degree 3) kernel
- LinearSVC (linear kernel)

regularization: $C=20.0$
Regularization $C=50$

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: $C=50.0$
Regularization $C=500$

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: $C=500.0$
Regularization $C=5000$

- SVC with linear kernel
- SVC with RBF kernel (gamma 0.7)
- SVC with polynomial (degree 3) kernel
- LinearSVC (linear kernel)

regularization: $C=5000.0$
Inseparable classes 12,13 (every 200)

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: C=1.0
Inseparable classes 12,13 (every 100)

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: C=1.0
Inseparable classes 12,13 (every 80)

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: C=1.0
Inseparable classes 12,13 (every 60)

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: C=1.0
Inseparable classes 12,13 (every 55)

SVC with linear kernel

SVC with RBF kernel (gamma 0.7)

SVC with polynomial (degree 3) kernel

LinearSVC (linear kernel)

regularization: $C=1.0$
Task and Homework kernels

- For PAMAP-Easy as divided into train\+test:
  - Cross-validate on train to choose between linear, poly and RBF.
  - Create the heatmap for RBF (i.e. plot score for all values of $C$ and gamma).
  - Use the GridSearchCV to find the best $C$ and gamma (i.e. find the best without plotting anything).
  - **NEVER USE THE test.txt FOR THE GRIDSEARCH**
  - Enter the accuracy of the best setting on test.txt.gz to CLASSIFICATION\_RESULTS.TXT, mention $C$ and gamma in the comment.

Due: May 8.