Identifying the native language (L1) of a writer based on a sample of their writing in a second language (L2)

**Our data**

- **L1s**: Arabic (ARA), Chinese (ZHO), French (FRA), German (DEU) Hindi (HIN), Italian (ITA), Japanese (JPN), Korean (KOR), Spanish (SPA), Telugu (TEL), Turkish (TUR)
- **L2**: English
- **Real-world objects**: For each L1, 1,000 texts in L2 from The ETS Corpus of Non-Native Written English (former TOEFL11), i.e. $\text{Train} \cup \text{DevTest}$
- **Target class**: L1

*More detailed info is available at the course website.*
Features used

96 numerical features = relative character frequencies

Example

"Finally having people with many academic broad know"

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.17073171</td>
<td>0.14634146</td>
<td>0.02439024</td>
<td>0.04878049</td>
<td>0.04878049</td>
</tr>
<tr>
<td>m</td>
<td>0.04878049</td>
<td>0.09756098</td>
<td>0.07317073</td>
<td>0.02439024</td>
<td>0.02439024</td>
</tr>
<tr>
<td>n</td>
<td>0.09756098</td>
<td>0.02439024</td>
<td>0.07317073</td>
<td>0.04878049</td>
<td>0.02439024</td>
</tr>
<tr>
<td>i</td>
<td>0.02439024</td>
<td>0.04878049</td>
<td>0.04878049</td>
<td>0.02439024</td>
<td>0.02439024</td>
</tr>
</tbody>
</table>
Support Vector Machines in R

Online demo
- Java applet at http://svm.dcs.rhbnc.ac.uk/

The implementation of SVMs in R
- `library(e1071)`, but there are also other libraries (`kernlab`, `shogun` ...)
- training: function `svm()`
- prediction: function `predict()`
- `svm()` can work in both classification and regression mode
- if response variable is categorical (factor) the engine switches to classification
model = svm(formula, data=, kernel=, cost=, cross=, ...)

- ?svm
- kernel defines the kernel used in training and prediction. The options are: linear, polynomial, radial basis and sigmoid (default: radial)
- cost – cost of constraint violation (default: 1)
- cross – optional, with the value k the k-fold cross-validation is performed
### SVM kernels in e1071

<table>
<thead>
<tr>
<th>Kernel name</th>
<th>Formula</th>
<th>Learning parameters and their default values</th>
</tr>
</thead>
<tbody>
<tr>
<td>linear</td>
<td>$x_i^T x_j$</td>
<td></td>
</tr>
<tr>
<td>polynomial</td>
<td>$(\gamma x_i^T x_j + c_0)^d$</td>
<td>$\gamma$, gamma=1/(data dimension) $c_0$, coef0=0 $d$, degree=3</td>
</tr>
<tr>
<td>radial</td>
<td>$\exp(-\gamma(|x_i - x_j|^2))$</td>
<td>$\gamma$, gamma=1</td>
</tr>
<tr>
<td>sigmoid</td>
<td>$\tanh(\gamma x_i^T x_j + c_0)$</td>
<td>$\gamma$, gamma=1/(data dimension) $c_0$, coef0=0</td>
</tr>
</tbody>
</table>
SVM – kernel functions

Non-linear kernel functions

• polynomial kernel
  – smaller degree can generalize better
  – higher degree can fit (only) training data better

• radial basis
  – very robust
  – you should try and use it when polynomial kernel is weak to fit your data
SVM Parameter tuning with `tune.svm`

- SVM is a more complicated method in comparison with the previous and usually requires parameter tuning!
- parameter tuning can take a very long time on big data, use a reasonably smaller part is often recommended

```r
> model.tune= tune.svm(class ~ ., data=train.small,
  kernel = "radial",
  gamma = c(0.001, 0.005, 0.01, 0.015, 0.02),
  cost = c(0.5, 1, 5, 10))
> model.tune
Parameter tuning of ‘svm’:

- sampling method: 10-fold cross validation

- best parameters:  
gamma  cost
  0.01  1

- best performance: 0.739
Built-in cross-validation

K-fold cross-validation

- parameter cross

```r
model.best <- svm(class ~ ., train.small,
                   kernel = "radial",
                   gamma = 0.01,
                   cost = 1,
                   cross = 10)

model.best$accuracies
[1] 33.0 27.5 31.0 33.5 28.0 29.0 29.0 33.5 33.0 34.5

model.best$tot.accuracy
# [1] 31.2

prediction.best <- predict(model.best, test, type="class")

mean(prediction.best==test$class)
[1] 0.3472727
```
Class weighting

- `class.weights` parameter
  In case of asymmetric class sizes you may want to avoid possibly overproportional influence of bigger classes. Weights may be specified in a vector with named components, like
  \[
  m \leftarrow \text{svm}(x, y, \text{class.weights} = c(A = 0.3, B = 0.7))
  \]
• Note that SVMs may be very sensible to the proper choice of parameters, so always check a range of parameter combinations, at least on a reasonable subset of your data.

• Be careful with large datasets as training times may increase rather fast.

• C-classification with the RBF kernel (default) can often be a good choice because of its good general performance and the few number of parameters (only two: cost and gamma).

• When you use C-classification with the RBF kernel: try small and large values for cost first, then decide which are better for the data by cross-validation, and finally try several gamma values for the better cost.