Twitter Feeds Profiling With TF-IDF

Juraj Petrik & Daniela Chuda
Task

- Given celebrity Twitter feed (English not guaranteed)
- Determine:
  - Fame level
  - Occupation
  - Age
  - Gender
Motivation

- Our background:
  - Source code authorship attribution – deep learning and frequency methods
  - Source code plagiarism detection – string similarity and character/word frequency methods
  - Useful in plagiarism and also source code – comments for example
Preprocessing

- Handles removal
- Same letters normalization
- URL replacing
- Emoji translation
- Dataset balancing
- Stop words removal
- Accent removal
- Lowercase
First approach

- Convolutional hierarchical recurrent NN
- Class imbalance problem – trained network tends to prefer majority class
  - Oversampling, synthetic, random – better, but not enough
  - Undersampling - little to no effect
- Another problem – variable length feeds and pretty long
- Custom loss function to reflect f1 score
- ...also painfully slow
- Result from testing dataset 1 is from this approach
Preprocessing

Handles removal
• @superuser ->

Same letters normalization
• faaaaancy -> fancy

URL filtering
• https://t.co/adsadasd -> URL_TOKEN
## Preprocessing

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Emoji translation</strong></td>
<td>• 😊 -&gt; :smiling face:</td>
</tr>
<tr>
<td><strong>Lowercase</strong></td>
<td>• AaaaA -&gt; aaaaa</td>
</tr>
<tr>
<td><strong>Accent removal</strong></td>
<td>• Čo sa deje -&gt; Co sa deje</td>
</tr>
<tr>
<td><strong>Stop words removal</strong></td>
<td>• The, on, an, a... -&gt;</td>
</tr>
</tbody>
</table>
Dataset balancing

- Random Oversampling
- SMOTE, TOMEK
Feature extraction

- N-gram based TF-IDF (1-3,5)
- Top 5000 features - grid search (matrix 5000x5000)
Classification

- One model per each “subtask”
- Random forest
- Extremely randomized trees
- Both have similar results, were more resistant to overfitting than our deep learning approaches
- Hyperparameter tuning – very similar results with 200+ trees
Regression

- Random forest regressor
- Used for birthyear trait
- Scaled to [0-1]
- Not so good in terms of the challenge as binning approaches
<table>
<thead>
<tr>
<th>Name</th>
<th>cRank</th>
<th>F1</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>gender</td>
<td>n</td>
</tr>
<tr>
<td>radivchev19</td>
<td>0.558</td>
<td>0.608</td>
<td>0.461</td>
</tr>
<tr>
<td>morenosandoval19</td>
<td>0.497</td>
<td>0.560</td>
<td>0.418</td>
</tr>
<tr>
<td>martinc19</td>
<td>0.465</td>
<td>0.594</td>
<td>0.485</td>
</tr>
<tr>
<td>fernquist19</td>
<td>0.412</td>
<td>0.465</td>
<td>0.300</td>
</tr>
<tr>
<td>petrik19</td>
<td><strong>0.440</strong></td>
<td><strong>0.555</strong></td>
<td><strong>0.385</strong></td>
</tr>
<tr>
<td>asif19</td>
<td>0.401</td>
<td>0.587</td>
<td>0.427</td>
</tr>
<tr>
<td>bryan19</td>
<td>0.230</td>
<td>0.335</td>
<td>0.165</td>
</tr>
<tr>
<td>Name</td>
<td>female</td>
<td>male</td>
<td>nonbinary</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------</td>
<td>-------</td>
<td>-----------</td>
</tr>
<tr>
<td>radivchev19</td>
<td>0.874</td>
<td>0.952</td>
<td>0</td>
</tr>
<tr>
<td>morenosandoval19</td>
<td>0.772</td>
<td>0.902</td>
<td>0</td>
</tr>
<tr>
<td>martinc19</td>
<td>0.835</td>
<td>0.943</td>
<td>0</td>
</tr>
<tr>
<td>fernquist19</td>
<td>0.449</td>
<td>0.866</td>
<td>0</td>
</tr>
<tr>
<td>petrik19</td>
<td>0.759</td>
<td>0.894</td>
<td>0</td>
</tr>
<tr>
<td>asif19</td>
<td>0.825</td>
<td>0.937</td>
<td>0</td>
</tr>
<tr>
<td>bryan19</td>
<td>0.014</td>
<td>0.838</td>
<td>0</td>
</tr>
</tbody>
</table>
Feature importance - fame
Feature importance - occupation
Possible improvements

- Oversampling – more sophisticated ones, focused on texts (synonyms, hypernyms from wordnet for example)
- Age prediction - regression vs bins (classification)
- Expand dataset – more data from Twitter (minority classes mainly)
- Language specific tuning