Product classification for e-Commerce platforms

Ioannis Partalas and Georgios Balikas

Viseo R&D, Laboratoire d'Informatique de Grenoble

January 27, 2016
Meetup, Grenoble Data Science

Outline

CDiscount competition

- Run on the `htpp://www.datascience.net` platform
- A large collection of product items were available
- Goal: classify new products to a product taxonomy
- Performance criterion: $\textit{Accuracy} = \frac{\#\text{products well classified}}{\#\text{total products}}$
- Prizes: 1st place 9,000 euros, 2nd 4,000e, 3rd 1,000e, 4th and 5th 500e
- Participated 175 teams. We were ranked 10th with score 64.2 (winning team had 68.3)

Accueil >> Cyclisme >> Equipements >> Housses de transport

NIKE Brassard pour téléphone portable Arm Band Diamond noir jaune

Réf : MP-44DE5M23522904

DESCRIPTION

Brassard pratique et confortable pour transporter votre Smartphone en tout simplicité

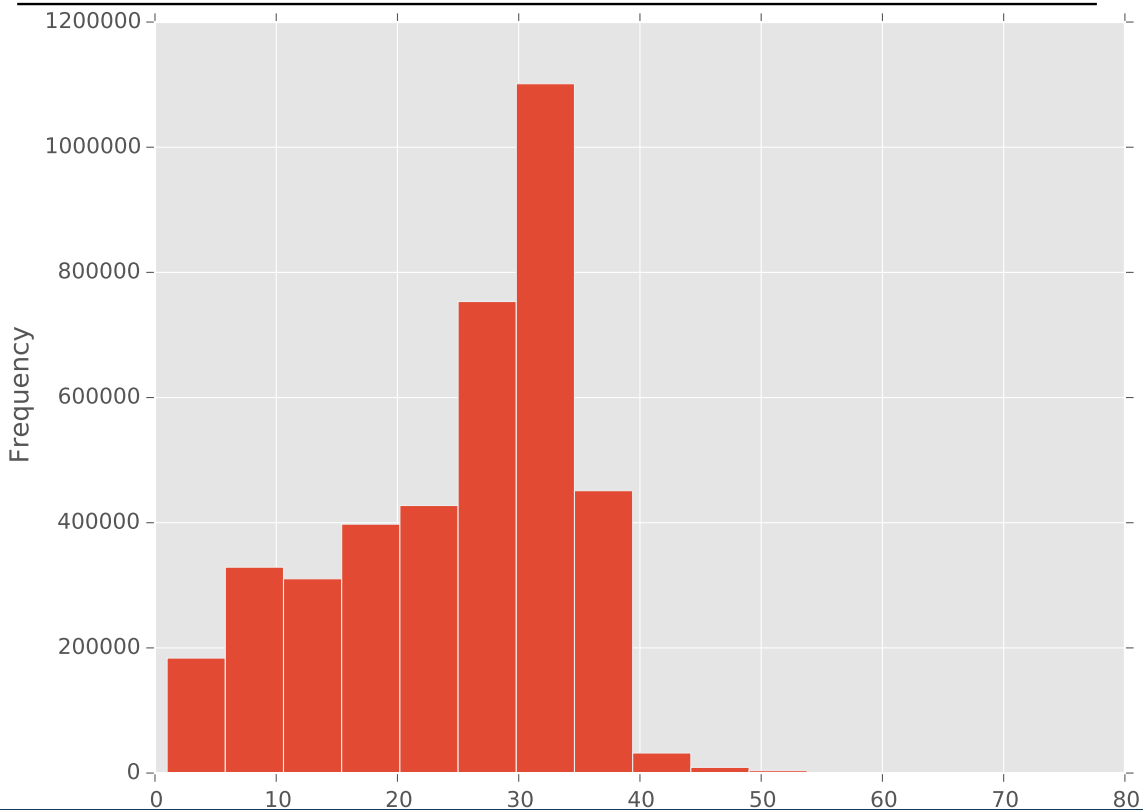
Product classification

- Critical task for e-commerce platforms (e.g. Amazon, e-Bay, Cdiscount, Kelkoo)
  - Supports retrieval and recommendation tasks
  - Shopping platforms use product taxonomies to this end
- Product classification can be framed as a text classification problem
  - $x_i \in R^d$ represents a document i in a vector space
  - $y_i \in \mathcal{Y} = \{1 \dots K\}$ its associated class label, $|\mathcal{Y}| > 2$
- Some problems
  - Titles are very short: "Lot de 20 pastilles de culture"
  - Problematic grammatical structure (incomplete sentences): "Pastis - Marseille - Vendu à l'unité"

Training Data

- ~15M products
- Hierarchy of classes with 3 levels: 52, 536 and 5,789 classes
- Target class: Categorie3
- 35,066 test instances

Categorie3	Description	Libelle	Marque
1000015309	De Collectif aux éditions SOLESMES	Benedictions de l eglise	
1000015309	Cartable 2 soufflets, compatible PC. Coloris : Rouge	Cartable	DELSEY
1000010100	or 750, poids : 3.45gr, diamants : 0.26carats	Bague or et diamants	AUCUNE
1000003407	Champagne Brut - Champagne-Vendu à l'unité-1 x 75cl	Mumm Brut	AUCUNE



Subsampling

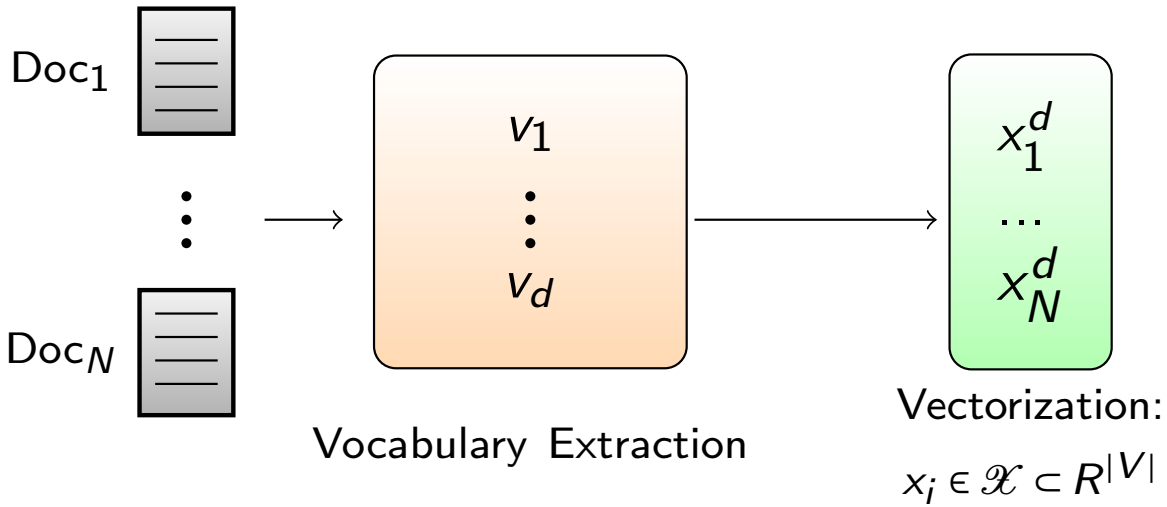
- Highly imbalanced dataset: models are biased towards big classes
- Data was randomly sampled by downsampling the majority classes
- Boosts around +2.5% the best single models
- Speeds up the training process

Preprocessing

- Concatenation of "Description"+"Libellé"+"Marque"
- Removal of non-ascii and non-printable characters, html tags and puncuations
- Accents were stripped
- Split words with numerical and text part: "12cm" → "12" and "cm"

```
def preprocess(text):
    t = re.sub('(<!--.*?-->|<[^>]*>)|([^[:print:]])|(_)',r' ',text)
    t = re.sub('\W',r' ',t)
    t = re.sub(r'([a-z]+)([0-9]+)',r'\1 \2',t)
    t = re.sub(r'([0-9]+)([a-z]+)',r'\1 \2',t)
    t = re.sub('[/><]',r' ',t)
    t = re.sub(r'\s+',r' ',t)
    return t;
```

Vectorization (1/2)

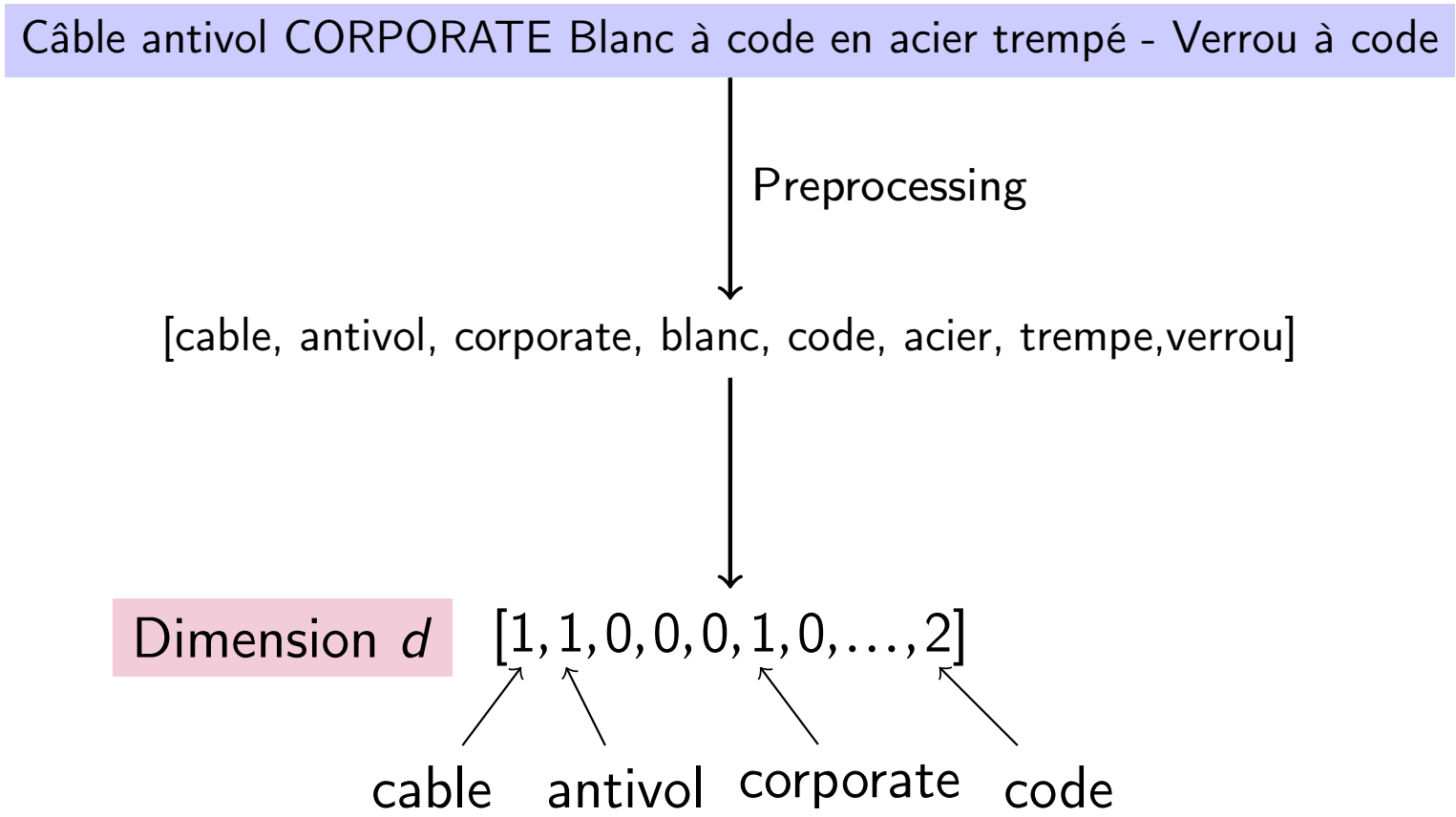


- Vocabulary extraction:
  - No stemming and lemmatization, we kept stop-words
  - Used several combinations of n-grams, $n = 1, 2, 3$
    - 2-grams for "Câble antivol CORPORATE Blanc à code" → (Câble antivol), (antivol CORPORATE), (CORPORATE Blanc), (Blanc code) ...
  - Indicatively $|V| = 1.6M$ for unigrams
  - We keep a fraction to reduce the problem

Vectorization (2/2)

- We adopted the Vector Space Model (Salton 1975):
  - $x_{ij} = tf_{v_j,i} \times idf_{v_j}$
  - $tf_{v_j,i}$ refers to the term frequency of term j in document i
  - $idf_v = log \frac{N}{df_v + 1}$, df_v is the document frequency of term v, N the total number or documents
  - Sublinear scaling $tf \leftarrow 1 + log(tf)$
- Each vector was normalized (unit vectors), $x_i \leftarrow x_i / ||x_i||$
- Power transformation: $x = (x_1, x_2, \ldots, x_d) \rightarrow (x_1^\alpha, x_2^\alpha, \ldots, x_d^\alpha)$
  - Reduces the effect of most common words
  - $\alpha = 0.5$ seems to work well

Vectorizing a document



Tuning and Validation Strategy

We used:

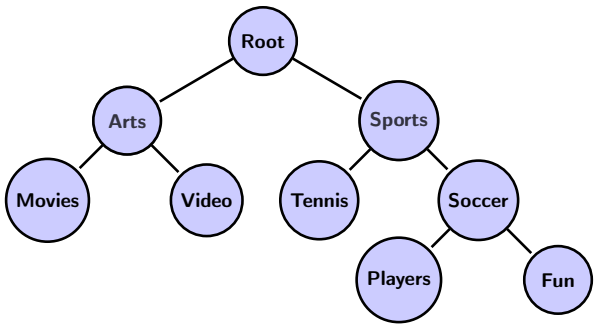
- a subset of classes to validate our ideas,
- the public part of the leaderboard to check our performance, and
- periodic rankings wrt to the private part to make sure we do not overfit the public part.

Models

- We rely on linear models focusing on SVMs

$$\min_w \frac{1}{2}||w||^2 + C\sum_i L(w; x_i, y_i)$$

- Loss functions: $\max(1 - y_i w^T x_i)$, $log(1 + e^{-y_i w^T x_i})$
- One-versus-rest for solving the multiclass problem
- We also employed several hierarchical top-down models
  - Keeping the whole structure
  - Removing layers from the hierarchy



Ensembling

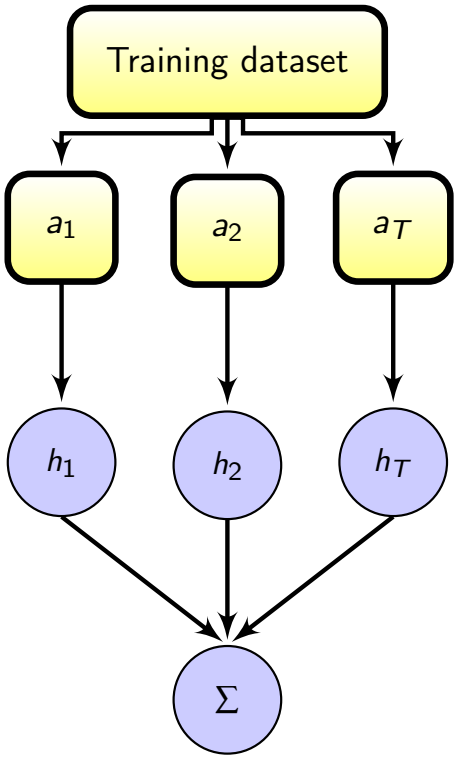
- Our final systems were combinations of the base models
- For an ensemble of classifiers $\{h_1, \ldots, h_T\}$
- Plurality voting

$$H(x) = c_{\arg\max_j \sum_{i=1}^{T} h_i^j(x)}$$

- Weighted voting

$$H(x) = c_{\arg\max_j \sum_{i=1}^{T} w_i h_i^j(x)}$$

- Unfortunately we had no time to try Stacked generalization



Hardware

- Full access to a machine with with 4 cores at 2.4Ghz and 16Gb of RAM
- Limited access to a machine with 24 cores at 3.3 Ghz and 128Gb of RAM
- Preprocessing takes around 1h + 4 to 6 hours for training a model

Base models results

	Description	Public	Private
1	200K unigrams	61.09	60.77
2	200K unigrams, α=0.5	61.60	61.25
3	250K unigrams	61.142	60.77
4	300K unigrams	61.148	60.87
5	250K unigrams, 250K bigrams	61.79	61.37
6	200K unigrams, 400K bigrams, α=0.5	62.09	61.76
7	200K unigrams, 400K bigrams, α=0.5, "Marque" as binary feature	62.64	62.15
8	1,2 M unigrams, bigrams, trigrams, α=0.5	62.28	61.99
9	2M unigrams, bigrams, trigrams	62.35	61.83
10	2M unigrams, bigrams, trigrams, α=0.5	63.30	62.99

- Power transformation boosts performance
- The addition of bigrams and trigrams helps. But may overfit the data

Best submissions

	Description	Public	Private	Coverage
1	270K unigrams, $\alpha = 0.5$, half data	63.56	63.11	3,208
2	Weighted voting 1	64.55	64.20	3,128
3	Weighted voting 2	64.57	64.14	3,116

- Downsampling improved the final score
- Weighted voting consistently improved accuracy by about 1.2%-1.8%
- Low coverage: 40% of the products in the training data belongs to 10 most common classes
- Best system in competition got 68.32%. We ranked 10th (we were in 1st place for over 1 month :( )

What didn't work

- k-Nearest Neighbors, Rochio (used mainly for ensembling)
- Distributed representations failed to improve the results
  - Used word2vec tool (Mikolov, 2013)
  - We generated a low-dimensional representation (200 features)
  - Improved k-NN classifiers over $tf - idf$ representation
- Tried also BM25 scheme instead of $tf - idf$ but results were worse

We explored also

- Sparsification of linear models (Moura et al., 2015)
  - Slight increase. Needs more investigation
  - No time to do further experiments
- Re-ranking for large-scale problems (Babbar et al., 2014)
  - Worked in some validations
  - Costly operation

Conclusions

- Knowing the problem helps the feature engineering process.
- The validation mechanism is of primary importance.
  - Do not trust the leaderboard
  - Make only a few submissions initially for testing the validation strategy
- High leaderboard ranks matter only after the end of the challenge.
- A clear strategy will benefit your participation in the long run.
- Published ideas do not apply universally.
- Ensembles always win.

Guidelines

- Learn your data. Try to understand them.
- Always have a validation strategy
- Do not fit leaderboard
- Keep always with you out-of-fold data
  - You may need them to stack, blend or validate
- Don't go for the money and don't do early dreams :)

Thank you

Questions?