April 3rd, 2015

PyData

Paris 2015

PriceMinister
Product catalog management with Scikit-Learn

Systematic
Paris Region Systems & ICT Cluster

Num FOCUS
Open Code, Better Science

PriceMinister Catalog Management

- 25+ Millions products, 150+ product types, 250+ categories
- Professionnal and individual sellers, many listing channels
- Machine Learning : a much discussed topic within PriceMinister / Rakuten, still discussing ...
- **And then came « Scikit-Learn »**
  - No data scientist background, little knowledge of python (django)
  - I Followed the very clear tutorial : « Working with text data »

UGC – User generated Classification

Machine Learning => Multi-class classification

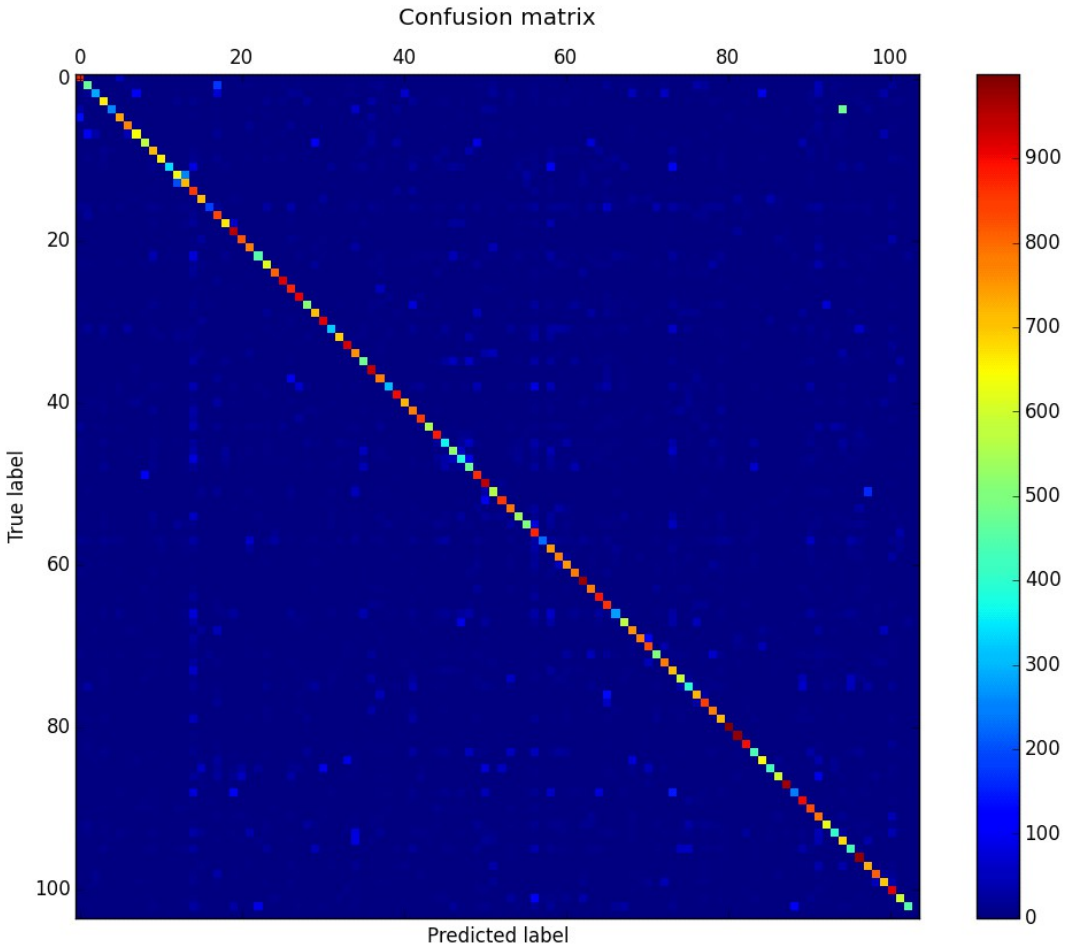
- TF-IDF: big USER dictionary (most discriminative words)
- Classifier: linear model (SGD)
- Labelling: products title already labelled on site by products type (good and bad classification)
- Training: 1000 samples per class (103 categories, no cultural products)
- Implementation : scikit-learn, scipy, numpy, pandas, Falcon, uwsgi (less than 30 lines of code)

UGC – User generated Classification

```
# -*- coding: utf-8 -*-

import falcon
import json
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import SGDClassifier
import numpy as np
import pandas as pd


class PredictResource(object):

    data = pd.read_csv('1000_each_type_3.txt',sep='\t', names=['pid','titre','type','pmattribute','type_freq'], encoding='utf-8')
    X = np.array(data.titre)
    y = np.array(data.type)
    tfidfvect = TfidfVectorizer()
    X_train_tfidf = tfidfvect.fit_transform(X)
    clf = SGDClassifier(loss='modified_huber', penalty='l2', alpha=1e-3, n_iter=5)
    clf.fit(X_train_tfidf, y)

    def on_get(self, req, resp):
        """Handles GET requests"""
        mareq = req.query_string
        prediction = {}
        reqvecteur = self.tfidfvect.transform([mareq])
        top3 = sorted(zip(self.clf.classes_, self.clf.predict_proba(reqvecteur)[0]), key=lambda x: x[1], reverse=True)[:3]
        for idx,t in enumerate(top3):
            prediction[idx+1] = u'%s (%.2f%%)' % (t[0], t[1] * 100)
        resp.status = falcon.HTTP_200
        resp.body = 'prediction(' + json.dumps(prediction) + ')'
```

First Results

mean accuracy of test : 0.67 => not that bad



Something is coming …

Clothes: Green Shirt

Stationery: Green Folder

Green Plant