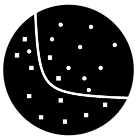
PROGETTO PER SISTEMI INTELLIGENTI

Prof. Nunzio Alberto Borghese
6 CFU

Luca Cappelletti

Project Documentation
Year 2017/2018



Magistrale Informatica
Università di Milano
Italy

23 settembre 2018

Indice

1 Introduction

The Zipf law is an empirical law that refers to the fact that in nature, many data can be approximated by a Zipfian distribution, for example texts, some images[1], even sounds in spoken languages[2]. It is therefore of interest to identify ways to exploit this relatively simple way to convert documents into representative vectors in problems such as classifications.

[1] https://www.dcs.warwick.ac.uk/bmvc2007/proceedings/CD-ROM/papers/paper-288.pdf

[2] https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0033993

2

The Zipf

The zipf function z converts any document $d \subseteq \mathbb{D}^m$ comprised of elements into a representative vector $\underline{\boldsymbol{v}} \subseteq [0,1]^n$, $n \leqslant m$ based on the frequency of said elements in the given document. Taken in consideration the set of elements $d \subseteq \mathbb{D}$ that comprise the document and $d_{\neq} \subseteq d$ the set of distinct elements of the set d, for any given element $d_{\neq_i} \in d_{\neq}$, the value assigned by the zipf function is:

$$z(d_{\neq_i}) : \mathbb{D} \to \mathbb{R} = \frac{\#\left\{\forall d_i \in d : d_{\neq_i} = d_i\right\}}{\#d}$$

3

The classifier

3.1 Training the classifier

3.1.1 Convert documents to vector representation

Given a set of training class-labeled elements T, we convert the documents, treated as *bag of elements*, to enumeration-sorted frequency vectors.

Given a document composed of $n = \#d$ elements $d = \{e_1, e_2, \dots, e_n\}$, first we define $d_{\neq} \subseteq d$ as the set of distinct elements in d. Then, we map to every distinct element its normalized arity in the set d:

$$z(e_i) = \frac{\#\left\{e_j \in d : e_j = e_i\right\}}{n}$$

Then, we proceed to map to a common enumeration the elements, using as full-set of elements the entire training set elements, so that $\forall e_i, e_j \in T, \exists! i, j \in \mathbb{N} : i = j \Leftrightarrow e_i = e_j$.

3.1.2 Choosing representative points

Given an arbitrary percentage of points $p \in [0,1]$ and an arbitrary distance in percentage $\alpha \in [0,1]$, for each class of points $C_j \in T$ we choose using k-Means:

$$k = \lceil \#C_j \cdot p^2 \rceil$$

This way the centroids surely distribute among the different points, following their density. If the points are, in truth, a unique cluster, the centroids will distribute themselves in an uniform fashion.

Then, we select for each cluster $Q_i \in \left\{Q_1, \dots, Q_k\right\}$ $\left\lfloor \#\left\{p \in C_j : p \in Q_i\right\} \cdot p \right\rfloor$ points on the border B_i. We move each point $p \in B_i$ of an amount proportional to the distance from the point to its centroid c_i:$\alpha \cdot L^2\left(p, c_i\right)$ towards its centroid.

Using a metric to choose k

For the high dimensionality and number of the vectors, iterating multiple times *KMeans* searching for an optimal k following any given metric has an high time cost. For this reason, an attempt using **PCA** to reduce the dimensionality and predict the number of clusters in high dimensionality space using a density metric was made.

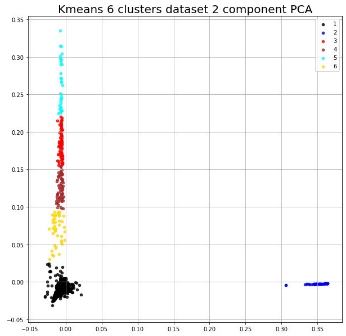
The density was defined as follows:

$$\overline{\rho}_{jk} = \frac{1}{k}\sum_{i=1}^{k} \rho_{jk_i} = \frac{1}{k}\sum_{i=1}^{k}\left(\frac{\#\left\{\underline{\boldsymbol{v}} \in C_{r_j} : \underline{\boldsymbol{v}} \in Q_i\right\}}{\#C_{r_j}}\right)^k \cdot \frac{1}{r_{Q_i}^2} \qquad r_{Q_i}^2 = \begin{cases} \frac{1}{n}\sum_{h=1}^{n}\left(\underline{\boldsymbol{c}}_i - \underline{\boldsymbol{p}}_h\right)^2 & \text{if } n \neq 0 \\ 1 & \text{else} \end{cases}$$

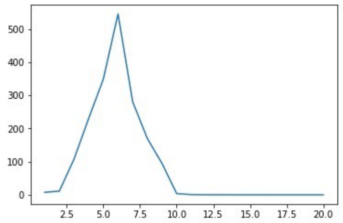
Where r_{Q_i} is the approximated radius of the cluster Q_i, using the farthest n frontier points p_f.

Does the prediction hold? While the metric on two dimensions reduction seemed to work, when iterated on increasingly larger number of dimensions, with the exception of strongly clustered data, it did not much better than a random selection.

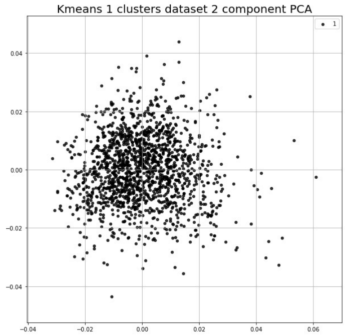
As follows, with 2D PCA density metrics the heuristic seems to be successful to identify the number of clusters k necessary to describe the given classes.



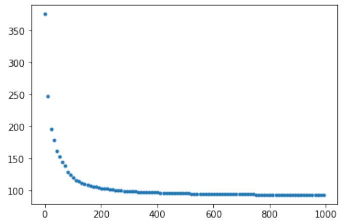
(a) Data with multi-clusters, 2D PCA, $k = 6$



(b) Density for data multi-clusters, 2D PCA, density/k



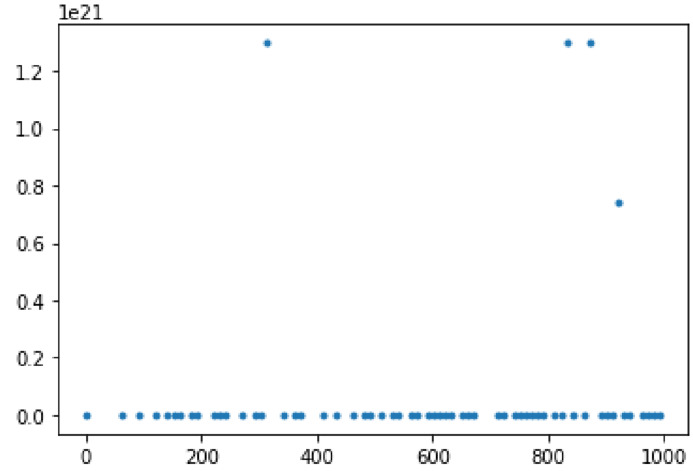
(c) Data with single-cluster, 2D PCA, $k = 1$



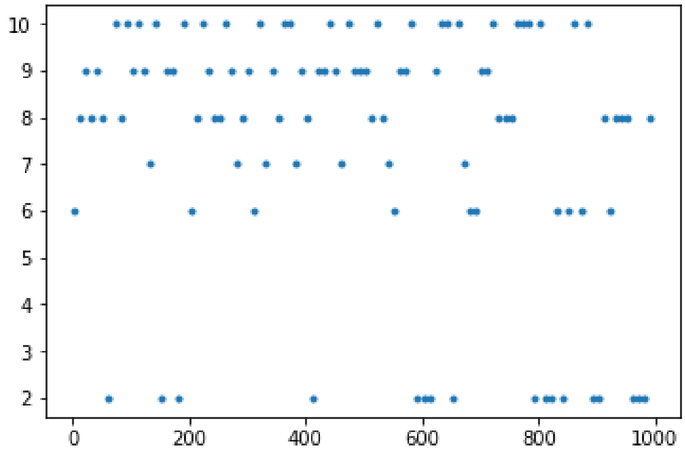
(d) Density for data single-cluster, 2D PCA, density/k

The prediction on any given PCA reduction however is not valid for different dimension numbers, on multi-clusters classes.

Any number of clusters k high enough ($k > 5$) is no more precise than a density metric. With increasing number of clusters, as shown below, it becomes increasingly unreliable.



(a) Best value of k for increasing dimensions in PCA reduction, one-clusters data.



(b) Best value of k for increasing dimensions in PCA reduction, multi-clusters data.

3.1.3 Completing the classifier

The classifier model is now finished, comprised of every class-labeled representative point.

3.2 Classifying a document

To classify a given a document d we proceed as follows:

1. Convert the document d to a zipf representative vector, using the common enumeration: $\underline{\boldsymbol{v}} = z(d)$.
2. The document is then classified with the same label as the closest representative point in the classifier model.