# Fast stepwise regression on linked data

Barbara Żogała-Siudem $^{1,2}$  and Szymon Jaroszewicz $^{2,3}$ 

 <sup>1</sup> Systems Research Institute, Polish Academy of Sciences Warsaw, Poland
 <sup>2</sup> Institute of Computer Science, Polish Academy of Sciences Warsaw, Poland
 s.jaroszewicz@ipipan.waw.pl
 <sup>3</sup> National Institute of Telecommunications Warsaw, Poland

Abstract. The main focus of research in machine learning and statistics is on building more advanced and complex models. However, in practice it is often much more important to use the right variables. One may hope that recent popularity of open data would allow researchers to easily find relevant variables. However current linked data methodology is not suitable for this purpose since the number of matching datasets is often overwhelming. This paper proposes a method using correlation based indexing of linked datasets which can significantly speed up feature selection based on classical stepwise regression procedure. The technique is efficient enough to be applied at interactive speed to huge collections of publicly available linked open data.

Keywords: stepwise feature selection, linked open data, spatial indexing

## 1 Introduction

It is well known from statistical modeling practice that including the right variables in the model is often more important than the type of model used. Unfortunately analysts have to rely on their experience and/or intuition as there are not many tools available to help with this important task.

The rising popularity of linked open data could offer a solution to this problem. The researcher would simply link their data with other variables downloaded from a public database and use them in their model. Currently, several systems exist which allow for automatically linking publicly available data ([2, 5, 11, 17, 18]). Unfortunately, those systems are not always sufficient. Consider, for example, a researcher who wants to find out which factors affect some variable available for several countries for several consecutive years. The researcher could then link publicly available data (from, say, Eurostat [1] or the United Nations [6]) by country and year to the target modeling variable and build a linear regression model using some method of variable selection. Unfortunately, such an approach is not practical since there are literally millions of variables available from Eurostat alone and most of them can be linked by country and year. As a result, several gigabytes of data would have to be downloaded and used for modeling.

This paper proposes an alternative approach: linking a new variable is performed not only by some key attributes but also based on the correlation with the target variable. We describe how to use spatial indexing techniques to find correlated variables quickly. Moreover, we demonstrate how such an index can be used to build stepwise regression models commonly used in statistics.

To the best of our knowledge no current system offers such functionality. The closest to the approach proposed here is the Google Correlate service [14]. It allows the user to submit a time series and find Google query whose frequency is most correlated with it. However Google Correlate is currently limited to search engine query frequencies and cannot be used with other data such as publicly available government data collections. Moreover it allows only for finding a single correlated variable, while the approach proposed here allows for automatically building full statistical models. In other words our contribution adds a statistical model construction step on top of a correlation based index such as Google Correlate.

There are several approaches to speeding up variable selection in stepwise regression models such as streamwise regression [22] or VIF regression [13]. None of them is, however, capable of solving the problem considered here: allowing an analyst to build a model automatically selecting from millions of available variables at interactive speeds.

Let us now introduce the notation. We will not make a distinction between a random variable and a vector of data corresponding to it. Variables/vectors will be denoted with lowercase letters  $x, y; \bar{x}$  is the mean of x and cor(x, y)correlation between x and y. Matrices (sets of variables) will be denoted with boldface uppercase letters, e.g. **X**. The identity matrix is denoted by **I** and **X**<sup>T</sup> is the transpose of the matrix **X**.

# 2 Finding most correlated variables. Multidimensional indexing

The simplest linear regression model we may think of is a model with only one variable: the one which is most correlated with the target. An example system building such models in the open data context is the Google Correlate tool [3, 14, 21]. It is based on the fact that finding a variable with the highest correlation is equivalent to finding a nearest neighbor of the response variable after appropriate normalization of the vectors.

In this paper we will normalize all input vectors (potential variables to be included in the model) as  $x' = \frac{x-\bar{x}}{\|x-\bar{x}\|}$ . That way, each vector is centered at zero and has unit norm, so we can think of them as of points on an (n-1)-sphere. It is easy to see that the correlation coefficient of two vectors x, y is simply equal

to the dot product of their normalized counterparts

$$\operatorname{cor}(x, y) = \langle x', y' \rangle.$$

Note that our normalization is slightly different from the one used in [14], but has the advantage that standard multidimensional indices can be used. After normalization the Euclidean distance between two vectors is directly related to their correlation coefficient

$$||x - y|| = \sqrt{2 - 2\operatorname{cor}(x, y)}.$$
(1)

The above equation gives us a tool to quickly find variables most correlated with a given variable, which is simply the one which is closest to it in the usual geometrical sense. Moreover to find all variables x whose correlation with y is at least  $\eta$  one needs to find all x's for which  $||x - y|| \leq \sqrt{2 - 2\eta}$ .

The idea now is to build an index containing all potential variables and use that index to find correlated variables quickly. Thanks to the relationship with Euclidean distance, multidimensional indexing can be used for the purpose. Building the index may be time consuming, but afterwards, finding correlated variables should be very fast. We now give a brief overview of the indexing techniques.

Multidimensional indexing. Multidimensional indices are data structures designed to allow for rapidly finding nearest neighbors in *n*-dimensional spaces. Typically two types of queries are supported. Nearest neighbor queries return k vectors in the index which are closest to the supplied query vector. Another type of query is range query which returns all vectors within a given distance from the query.

Due to space limitations we will not give an overview of multidimensional techniques, see e.g. [19]. Let us only note that Google Correlate [21] uses a custom designed technique called Asymmetric Hashing [8].

In the current work we use Ball Trees [12] implemented in the Python Scikit.Learn package. Ball Trees are supposed to work well even for high dimensional data and return exact solutions to both nearest neighbor and range queries. For faster, approximate searches we use the randomized kd-trees implemented in the FLANN package [15] (see also [16]).

Of course finding most correlated variables has already been implemented by Google Correlate. In the next section we extend the technique to building full regression models, which is the main contribution of this paper.

## 3 Stepwise and stagewise regression

In this section we will describe classical modeling techniques: stagewise and stepwise linear regression and show how they can be efficiently implemented in the open data context using a multidimensional index.

Stagewise regression is a simple algorithm for variable selection in a regression model which does not take into account interactions between predictor variables, see e.g. [10] for a discussion. The algorithm is shown in Figure 1. The idea is simple: at each step we add the variable most correlated with the residual of the current model. The initial residual is the target variable y and the initial model matrix  $\mathbf{X}$  contains just a column of ones responsible for the intercept term. The matrix  $\mathbf{H}_{\mathbf{X}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$  is the projection matrix on  $\mathbf{X}$ , see [9] for details.

Algorithm: Stagewise

1) $r \leftarrow y; \mathbf{X} \leftarrow (1, 1, \dots, 1)^T,$
2) Find a variable $x_i$ most correlated with $r$ ,
3) Add $x_i$ to the model: $\mathbf{X} \leftarrow [\mathbf{X} x_i]$ ,
4) Compute the new residual vector $r = y - \mathbf{H}_{\mathbf{X}} y$
5) If the model has improved: goto 2.

Fig. 1. The stagewise regression algorithm.

The stopping criterion in step 5 is based on the residual sum of squares:  $RSS = r^T r = ||r||^2$ , where r is the vector of residuals (differences between true and predicted values). The disadvantage of RSS is that adding more variables can only decrease the criterion. To prevent adding too many variables to the model additional penalty terms are included, the two most popular choices are Akaike's AIC [4] and Schwarz's BIC [20]. Here we simply set a hard limit on the number of variables included in the model.

The advantage of stagewise regression is its simplicity, one only needs to compute the correlation of all candidate variables with the residual r. Thanks to this, one can easily implement stagewise regression using techniques from Section 2, so the approach can trivially be deployed in the proposed setting.

The disadvantage of stagewise regression is that is does not take into account correlations between the new variable and variables already present in the model. Consider an example dataset given in Table 1. The dataset has three predictor variables  $x_1, x_2, x_3$  and a target variable y. The data follows an exact linear relationship:  $y = 3x_1 + x_2$ . It can be seen that the variable most correlated with y is  $x_1$ , which will be the first variable included in the model. The residual vector of that model, denoted  $r_1$ , is also given in the table. Clearly the variable most correlated with  $r_1$  is  $x_3$  giving a model  $y = \beta_0 + \beta_1 x_1 + \beta_2 x_3$ . But the true model does not include  $x_3$  at all! The reason is that  $x_3$  is highly correlated with  $x_1$ , and this correlation is not taken into account by stagewise regression.

Table 1. Example showing the difference between stepwise and stagewise regression.

$x_1$	$x_2$	$x_3$	y	$r_1$
0.03	-0.12	0.75	-0.03	0.51
-0.54	-0.10	-0.47	-1.71	-0.15
0.13	-1.03	0.11	-0.64	-0.27
0.73	-1.58	0.00	0.61	-0.09

An improvement on stagewise regression is *stepwise regression* proposed in 1960 by Efroymson [7]. The algorithm is given in Figure 2. The main idea is that at each step we add each variable to the model, compute the actual residual sum of squares (which takes into account correlations between variables) and add the variable which gives the best improvement.

Algorithm: Stepwise

1)	$r \leftarrow y; \mathbf{X} \leftarrow (1, 1, \dots, 1)^T,$
2)	For each variable $x_i$ :
	compute the residual of the model obtained by adding $x_i$ to the current model:
	$r_i = y - \mathbf{H}_{[\mathbf{X} x_i]} y$
3)	Find $x_{i^*}$ , where $i^* = \arg\min r_i^T r_i$ ,
4)	If model: $[\mathbf{X} x_{i^*}]$ is better than $\mathbf{X}$ :
	1. Add $x_{i^*}$ to the model $\mathbf{X} \leftarrow [\mathbf{X} x_{i^*}]$
	2. goto 2).

Fig. 2. The stepwise regression algorithm.

In the example stepwise regression will choose the correct variables  $x_1$  and then  $x_2$ , which is the best possible model. In general, stepwise regression builds better models than stagewise regression, but is more costly computationally. At each step we need to compute the RSS for several regression models, which is much more expensive than simply computing correlations.

## 4 Fast stepwise selection based on multidimensional indices

Stepwise regression is known to give good predictions, however when the number of attributes is large, it becomes inefficient; building a model consisting of many variables when we need to search through several millions of candidates, as is often the case with linked data, would be extremely time consuming, since at each step we would need to compute RSS of millions of multidimensional models.

In this section we present the main contribution of this paper, namely an approach to speed up the process using a multidimensional index. Our goal is to decrease the number of models whose RSS needs to be computed at each step through efficient filtering based on a multidimensional index. Assume that k-1 variables are already in a model and we want to add the k-th one. Let  $\mathbf{X}_{k-1}$  denote the current model matrix. The gist of our approach is given in the following theorem.

**Theorem 1.** Assume that the variables  $x_1, \ldots, x_{k-1}$  currently in the model are orthogonal, i.e.  $\mathbf{X}_{k-1}^T \mathbf{X}_{k-1} = \mathbf{I}$  and let  $r = y - \mathbf{H}_{\mathbf{X}_{k-1}}y$  denote the residual vector of the current model. Consider two variables  $x_k$  and  $x_{k'}$ . Denote  $c_{i,k} = cor(x_i, x_k), c_{i,k'} = cor(x_i, x_{k'}), c_{r,k} = cor(r, x_k), c_{r,k'} = cor(r, x_{k'})$ . Let  $\mathbf{X}_k = [\mathbf{X}_{k-1}|x_k]$  and  $\mathbf{X}_{k'} = [\mathbf{X}_{k-1}|x_{k'}]$ . Further, let  $r_k = y - \mathbf{H}_{\mathbf{X}_k}y$  be the residual

vector of the regression model obtained by adding variable  $x_k$  to the current model, and let  $r_{k'}$  be defined analogously. Then,  $||r_{k'}||^2 \leq ||r_k||^2$  implies

$$\max\left\{|c_{1,k'}|,\ldots,|c_{k-1,k'}|,|c_{r,k'}|\right\} \ge \frac{|c_{r,k}|}{\sqrt{1-c_{1,k}^2-\ldots-c_{k-1,k}^2+(k-1)c_{r,k}^2}}.$$
 (2)

The theorem (the proof can be found in the Appendix) gives us a way to implement a more efficient construction of regression models through the stepwise procedure. Each step is implemented as follows. We first find a variable  $x_k$  which is most correlated with the current residual r. Then, using the right hand side of Equation 2 we find the lower bound for correlations of the potential new variable with the current residual and all variables currently in the model. Then, based on Equation 1, we can use k range queries (see Section 2) on the spatial index to find all candidate variables. Steps 2 and 3 of Algorithm 2 are then performed only for variables returned by those queries. Since step 2 is the most costly step of the stepwise procedure this can potentially result in huge speedups. The theorem assumes  $x_1, \ldots, x_{k-1}$  to be orthogonal which is not always the case. However we can always orthogonalize them before applying the procedure using e.g. the QR factorization.

The final algorithm is given in Figure 3. It is worth noting that (when exact index is used like the Ball Tree) algorithm described in Figure 3 gives the same results as stepwise regression performed on full, joined data.

#### Algorithm: Fast stepwise based on multidimensional index

- 1)  $r \leftarrow y; \mathbf{X} \leftarrow (1, 1, \dots, 1)^T$
- 2) Find a variable  $x_1$  most correlated with r # nearest neighbor query
- 3) Add  $x_1$  to the model:  $\mathbf{X} \leftarrow [\mathbf{X}|x_1]$
- 4) Compute the new residual vector  $r = y \mathbf{H}_{\mathbf{X}} y$
- 5) Find a variable  $x_k$  most correlated with r
- 6)  $C \leftarrow \{x_k\} \#$  the set of candidate variables
- 7)  $\eta \leftarrow \frac{|c_{r,k}|}{\sqrt{1 c_{1,k}^2 \dots c_{k-1,k}^2 + (k-1)c_{r,k}^2}}$
- 8) **For**  $i \leftarrow 1, ..., k 1$ :
- 9)  $C \leftarrow C \cup$  all variables x such that  $||x x_i||^2 \leq 2 2\eta \#$  range queries
- 10)  $C \leftarrow C \cup$  all variables x such that  $||r x_i||^2 \leq 2 2\eta \#$  range query
- 11) Find the best variable  $x_{i^*}$  in C using stepwise procedure, add it to the model 12) If the model has improved significantly: goto 4).

Fig. 3. The fast stepwise regression algorithm based on multidimensional index.

## 5 Experimental evaluation

We will now present an experimental evaluation of the proposed approach. First we give an illustrative example, then examine the efficiency.

#### 5.1 An illustrative example

The following example is based on a part of the Eurostat database [1]. The response variable is the infant mortality rate in each country and the predictors are variables present in a part of the database concerning 'Population and social conditions', mainly 'Health'. The combined data set consists of 736 observations (data from 1990 till 2012 for each of the 32 European countries) and 164460 variables. We decided to select two variables for the model. Missing values in the time series were replaced with the most previous available value or with the next one if the previous did not exist.

Exact stepwise regression (produced with regular stepwise procedure or the Ball Tree index) resulted in the following two variables added to the model:

- "Causes of death by NUTS 2 regions Crude death rate (per 100,000 inhabitants) for both men an women of age 65 or older, due to Malignant neoplasms, stated or presumed to be primary, of lymphoid, haematopoietic and related tissue"
- ,,Health personnel by NUTS 2 regions number of physiotherapists per inhabitant".

The variables themselves are not likely to be directly related to the target, but are correlated with important factors. The first variable is most probably correlated with general life expectancy which reflects the efficiency of the medical system. The number of physiotherapists (second variable) is most probably correlated with the number of general health personnel per 100,000 inhabitants. Dealing with correlated variables is an important topic of the future research.

An analogous result was obtained using an approximate index implemented in the FLANN package. Due to the fact that the results are approximated, slightly different attributes were selected but the RSS remained comparable. Moreover, building the model using the Ball Tree index was almost 8 times faster than stepwise regression on full data, and using the FLANN index more than 400 times faster!

#### 5.2 Performance evaluation

To assess performance we used a part of the Eurostat [1] database. The response variable was again the infant mortality rate and predictors came from the 'Population and social conditions' section, mainly: 'Health', 'Education and training' and 'Living conditions and welfare'. This resulted in a joined dataset consisting of 736 observations (data from 1990 till 2012 for 32 European countries) and over 200,000 attributes.

The algorithms used in comparison are regular stepwise regression on full joined data ('regular step'), fast stepwise regression using two types of spatial indices and stepwise regression built using the algorithm in Figure 3 with spatial queries answered using brute force search ('step with no index').

The first two charts in Figure 4 show how the time to build a model with 3 or 5 variables changes with growing number of available attributes (i.e. the



Fig. 4. Average time needed to build models with 3 or 5 variables for varying numbers of available variables and observations. 'regular step' is the classical stepwise regression, all others are the proposed fast versions using different spatial indexes and brute search.

size of full joined data). The second two charts show how the time changes with growing number of observations (records). To obtain the smaller datasets we simply drew samples of the attributes or of the observations. We can see that the best times can be obtained using FLANN index. It is worth noting that FLANN gives approximate, yet quite precise results. Slower, but still reasonably fast model construction can be obtained by using Ball Tree index, which guarantees the solution is exact. All charts show that the bigger the data the bigger the advantage from using the algorithm shown in Figure 3.

## 6 Conclusions

The paper presents a method for building regression model on linked open data at interactive speeds. The method is based on the use of spatial indexes for efficient finding of candidate variables. The method has been evaluated experimentally on Eurostat data and demonstrated to perform much faster than standard regression implementations.

## 7 Acknowledgements

The paper is co-founded by the European Union from resources of the European Social Fund. Project PO KL ,,Information technologies: Research and their interdisciplinary applications", Agreement UDA-POKL.04.01.01-00-051/10-00.

## A Proof of Theorem 1

To prove Theorem 1 we need to introduce two lemmas.

**Lemma 1.** Adding  $x_k$  to a least squares model of y based on  $\mathbf{X}_{k-1} = [x_1| \dots |x_{k-1}]$ decreases the RSS by  $\frac{(x_k^T \mathbf{P}_{k-1} y)^2}{x_k^T \mathbf{P}_{k-1} x_k}$ , where  $\mathbf{P}_{k-1} := \mathbf{I} - \mathbf{H}_{\mathbf{X}_{k-1}}$ .

*Proof.*  $x_k$  can be expressed as a sum of vectors in the plane spanned by  $\mathbf{X}_{k-1}$ and perpendicular to that plane:  $x_k = \mathbf{H}_{\mathbf{X}_{k-1}}x_k + \mathbf{P}_{k-1}x_k$ . If  $x_k$  is a linear function of columns of  $\mathbf{X}_{k-1}$ , adding it to the model gives no decrease of RSS, so we only need to consider  $\mathbf{P}_{k-1}x_k$ . It is easy to see that if  $x_k$  is uncorrelated with each column of  $\mathbf{X}_{k-1}$ , adding it to the model decreases RSS by  $\frac{(x_k^T y)^2}{x_k^T x_k}$ . This is because the RSS is then equal to  $y^T \mathbf{P}_k y$ , where  $\mathbf{P}_k = \mathbf{P}_{k-1} - x_k (x_k^T x_k)^{-1} x_k^T$ . Combining those facts with symmetry and idempotency of  $\mathbf{P}_{k-1}$ , RSS decreases by

$$\frac{((\mathbf{P}_{k-1}x_k)^T y)^2}{(\mathbf{P}_{k-1}x_k)^T \mathbf{P}_{k-1}x_k} = \frac{(x_k^T \mathbf{P}_{k-1}^T y)^2}{x_k^T \mathbf{P}_{k-1}^T \mathbf{P}_{k-1}x_k} = \frac{(x_k^T \mathbf{P}_{k-1}y)^2}{x_k^T \mathbf{P}_{k-1}x_k}$$

**Lemma 2.** Assume now that  $\mathbf{X}_{k-1}$  is orthogonal. If adding  $x_{k'}$  to the model gives lower RSS than adding  $x_k$ , then:

$$\frac{c_{r,k}^2}{1 - c_{1,k}^2 - \dots - c_{k-1,k}^2} < \frac{c_{r,k'}^2}{1 - c_{1,k'}^2 - \dots - c_{k-1,k'}^2}.$$
(3)

*Proof.* From Lemma 1 we know that if  $x_{k'}$  causes greater decrease in RSS then

$$\frac{(x_k^T \mathbf{P}_{k-1} y)^2}{x_k^T \mathbf{P}_{k-1} x_k} < \frac{(x_{k'}^T \mathbf{P}_{k-1} y)^2}{x_{k'}^T \mathbf{P}_{k-1} x_{k'}}$$

We also know that (since vectors are normalized)  $c_{r,k}^2 = (x_k^T \mathbf{P}_{k-1} y)^2$ , and using orthogonality of  $\mathbf{X}_{k-1}$  we get

$$x_k^T \mathbf{P}_{k-1} x_k = x_k^T (\mathbf{I} - \mathbf{X}_{k-1} (\mathbf{X}_{k-1}^T \mathbf{X}_{k-1})^{-1} \mathbf{X}_{k-1}^T) x_k = x_k^T (\mathbf{I} - \mathbf{X}_{k-1} \mathbf{X}_{k-1}^T) x_k = x_k^T x_k - (x_1^T x_k)^2 - \dots - (x_{k-1}^T x_k)^2 = 1 - c_{1,k}^2 - \dots - c_{k-1,k}^2,$$

which proves the lemma.

Proof (of Theorem 1). If for any i = 1, ..., k-1:  $|c_{i,k'}| \ge \frac{|c_{r,k}|}{\sqrt{1-c_{1,k}^2-...-c_{k-1,k}^2+(k-1)c_{r,k}^2}}$  then the inequality is true. Otherwise for all i = 1, ..., k-1:

$$|c_{i,k'}| < \frac{|c_{r,k}|}{\sqrt{1 - c_{1,k}^2 - \dots - c_{k-1,k}^2 + (k-1)c_{r,k}^2}}$$
(4)

and we need to show that this implies  $|c_{r,k'}| \ge \frac{|c_{r,k}|}{\sqrt{1-c_{1,k}^2-\ldots-c_{k-1,k}^2+(k-1)c_{r,k}^2}}$ . Notice first that the inequalities (4) imply

$$1 - c_{1,k'}^2 - \ldots - c_{k-1,k'}^2 > \frac{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2}{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2 + (k-1)c_{r,k}^2}$$

Using this inequality and Lemma 2 we get the desired result:

$$c_{r,k'}^2 \geqslant c_{r,k}^2 \frac{1 - c_{1,k'}^2 - \ldots - c_{k-1,k'}^2 + c_{r,k'}^2}{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2 + c_{r,k}^2} > \frac{c_{r,k}^2}{1 - c_{1,k}^2 - \ldots - c_{k-1,k}^2 + (k-1)c_{r,k}^2}$$

### References

- 1. Eurostat. http://ec.europa.eu/eurostat.
- 2. Fegelod. http://www.ke.tu-darmstadt.de/resources/fegelod.
- 3. Google correlate. http://www.google.com/trends/correlate.
- H. Akaike. A new look at the statistical model identification. *IEEE Transactions* on Automatic Control, AC-19(6):716–723, 1974.
- C. Bizer, T. Heath, and T. Berners-Lee. Linked data the story so far. International Journal on Semantic Web and Information Systems (IJSWIS), 5(3):1–22, 2009.
- 6. United Nations Statistics Division. Undata. http://data.un.org.
- 7. M. A. Efroymson. Multiple Regression Analysis. Wiley, 1960.
- A. Gersho and R. M. Gray. Vector Quantization and Signal Compression. Springer, 1992.
- 9. T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.
- T. Hastie, R. Tibshirani, and J. Friedman. The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer-Verlag, 2009.
- 11. T. Heath and Bizer C. *Linked Data: Evolving the Web into a Global Data Space.* Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool, 1 edition, 2011.
- A. M. Kibriya and E. Frank. An empirical comparison of exact nearest neighbour algorithms. In *PKDD*, pages 140–151. Springer, 2007.
- D. Lin and D. P. Foster. Vif regression: A fast regression algorithm for large data. In ICDM '09. Ninth IEEE International Conference on Data Mining, 2009.
- M. Mohebbi, D. Vanderkam, J. Kodysh, R. Schonberger, H. Choi, and S. Kumar. Google correlate whitepaper. 2011.
- M. Muja and D. Lowe. FLANN Fast Library for Approximate Nearest Neighbors, 2013.
- 16. M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36, 2014.
- H. Paulheim. Explain-a-lod: using linked open data for interpreting statistics. In *IUI*, pages 313–314, 2012.
- H. Paulheim and J. Fürnkranz. Unsupervised generation of data mining features from linked open data. Technical Report TUD-KE-2011-2, Knowledge Engineering Group, Technische Universität Darmstadt, 2011.
- H. Samet. The Design and Analysis of Spatial Data Structures. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1990.
- G. Schwarz. Estimating the dimension of a model. The Annals of Statistics, 6(2):461–464, 1978.
- 21. D. Vanderkam, R. Schonberger, H. Rowley, and S. Kumar. Technical report: Nearest neighbor search in google correlate. 2013.
- J. Zhou, D. P. Foster, R. A. Stine, and L. H. Ungar. Streamwise feature selection. J. Mach. Learn. Res., 7:1861–1885, December 2006.