Detecting Bad Wikipedia Edits

Brett Kuprel
December 14, 2012

Introduction

Wikipedia is often criticized for having inaccurate information. It would be nice if potentially bad edits were 1) automatically flagged, and 2) brought to the attention of users more acquainted with the area. Consider the following (simplified) scenario:

> *Amy edits mostly physics articles. Bob edits mostly history articles. Amy edits a history article. The edit is flagged and brought to Bob's attention.*

I present a method for detecting and managing these anomalous edits.

Data

A week's worth of Wikipedia edits were collected from August 30, 2010 to September 6, 2010 in the following form:

> [User ID] [Article ID] [Time of Edit]

I save the last 10% of data (about a day's worth) to test on. After removing articles edited less than 5 times, users who made less than 5 edits, and multiple edits by a user to an article, the data represents $|E| = 16{,}506$ edits to $n = 2{,}077$ articles made by $m = 1{,}205$ users.

Matrix Representation

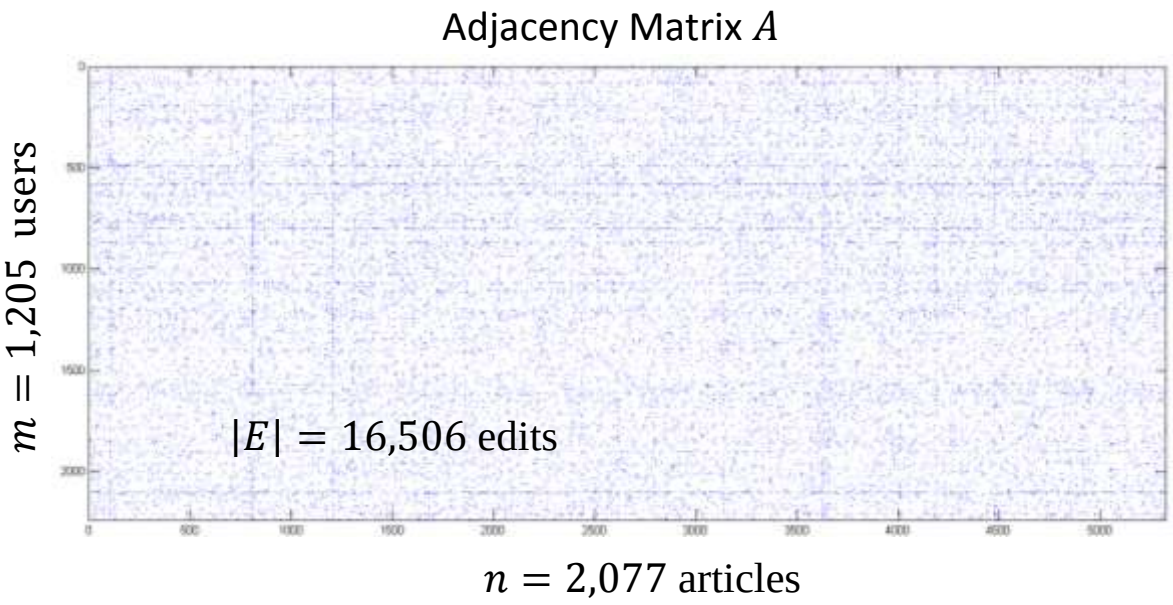
I reorganize the data into a sparse matrix

$$A_{ij} = \begin{Bmatrix} 1, \text{user } i \text{ edited article } j \\ 0, \text{otherwise} \end{Bmatrix}$$

Using the following code in Matlab

```
load('DATA.mat');
DATA=sortrows(DATA,3);
data=DATA(:,[1 2]);
[~,ia,~]=unique(data,'rows');
data=data(sort(ia),:);
data_train=data(1:round(.9*end),:);
data_test=data(round(.9*end)+1:end,:);
m=max(data_train(:,1)); n=max(data_train(:,2));
vet=and(data_test(:,1)<=m,data_test(:,2)<=n);
data_test=data_test(vet,:);
A=sparse(data_train(:,1),data_train(:,2),1,m,n);
A_test=sparse(data_test(:,1),data_test(:,2),1,m,n);
editmin=5;
while sum(sum(A,1)<editmin) || sum(sum(A,2)<editmin)
    x=sum(A,2)>=editmin;
    y=sum(A,1)>=editmin;
    A=A(x,y);
    A_test=A_test(x,y);
end
[m n]=size(A);
```

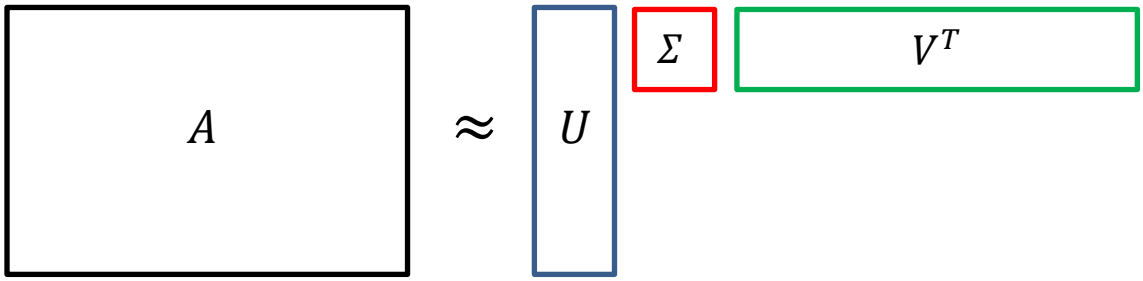
The resulting matrix representation of the data looks like this:



The data is much more sparse than it appears, only 0.27% of A's elements are nonzero.

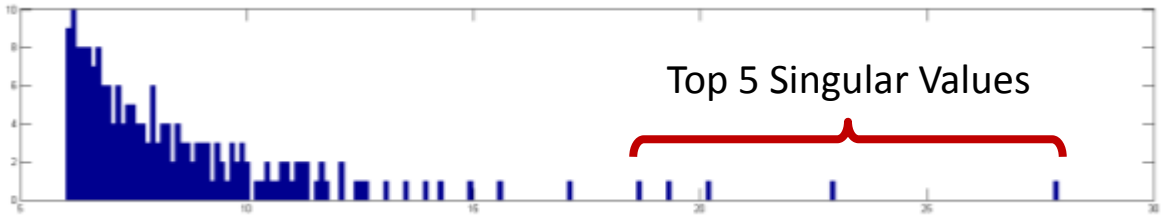
Model

The adjacency matrix A can be modeled by a low rank matrix using singular value decomposition (SVD).




```
[U S V]=svds(A,k);
```

Where k is the rank of the approximation. Looking at a histogram of the top 200 singular values of A, it appears 5 is a reasonable choice for k.



Data/Parameter Analysis

It is important to not overfit the data.

Number of users,	$m = 1{,}205$
Number of articles,	$n = 2{,}077$
Number of features,	$k = 5$
Number of edits,	$N = 16{,}506$
Ratio of edits to parameters	$\frac{N}{(m+n)k} \approx 1.006$

There is about 1 data point for each parameter. Could be better, but it is reasonable.

Features

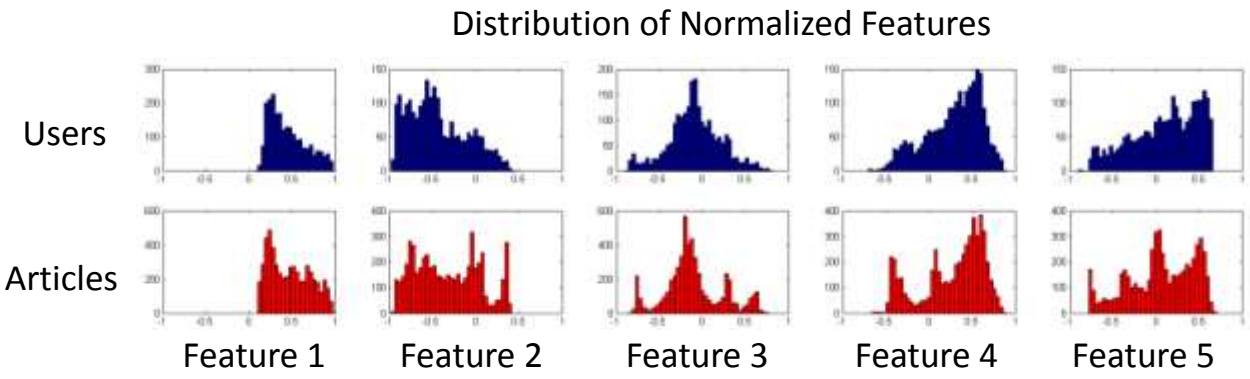
The rows of U and V can be thought of as feature vectors for articles and users. An element of A is then represented by the dot product between the editing user's feature vector and the article's feature vector.

$$\left.\begin{matrix}(U\sqrt{S})^T = [u_1 & \cdots & u_m],\ u_i \in \mathbb{R}^k \\ (V\sqrt{S})^T = [v_1 & \cdots & v_n],\ v_j \in \mathbb{R}^k\end{matrix}\right\} \rightarrow A_{ij} = \langle u_i, v_j \rangle$$

I normalize the feature vectors so that the inner products are between -1 and 1.

$$u_i := \frac{u_i}{|u_i|},\ v_j := \frac{v_j}{|v_j|}$$

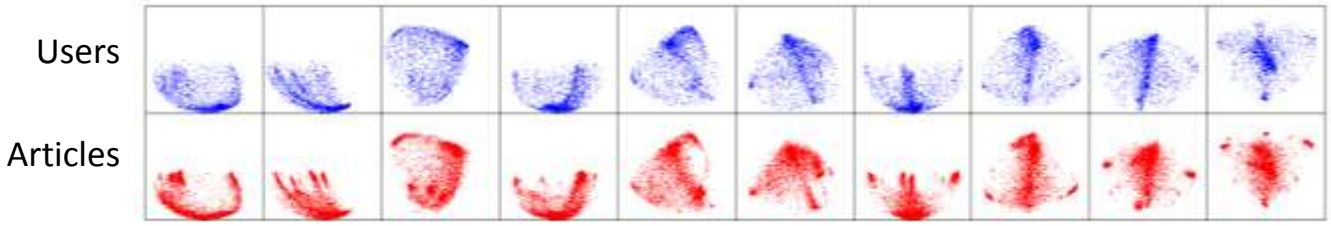
Let me plot distributions of the features for the Wikipedia data.



They look similar, this is good. I also want to visualize the locus of points in feature space. I can't make a $k = 5$ dimensional plot, but I can plot ${}_5C_2 = 10$ planar projections. To best see the structure in the data, I will plot along the principal axes using PCA (*I don't subtract the mean or divide by the standard deviations because the feature vectors would no longer be unit length)

```
u=normr(U*sqrt(s));
v=normr(V*sqrt(s));
[Q,~]=eigs([u;v]'*[u;v],k);
u=u*Q;
v=v*Q;
```

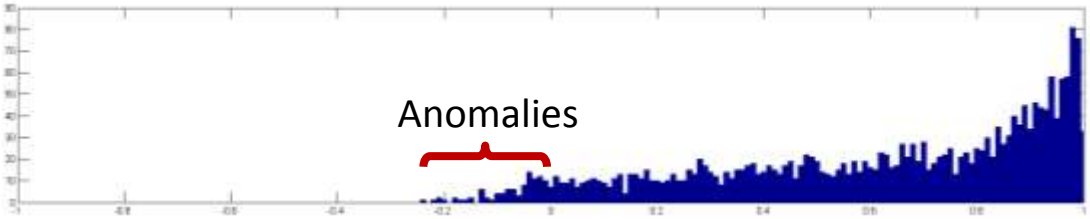
Principal Cross Sections of Normalized Features $\in \mathbb{R}^{k=5}$



It is clear from the planar projections that the locus of articles in feature space is very similar to the locus of users. This implies that physicists edit physics articles, history people edit history articles, etc[1].

Anomaly Detection

The normalized feature vectors lie on a k-dimensional sphere. Similiarity is then easily expressed as an inner product between user and article feature vectors. A dot product of -1 represents unrelated feature vectors, whereas a dot product of 1 represents related feature vectors. The negative dot prodcuts correspond to anomalous edits (e.g. an arts major editing a biology article).



$\langle u_i, v_j \rangle$

Feedback

These anomalous edits should be reported to users with feature vectors more aligned with those articles. The more experienced user would then mark the edit as good or bad. This rating would affect the less experienced user's quality score, i.e. it could have just been a grammar edit, no harm done.

[1]To see this, imagine that there were only two articles on Wikipedia, then there would be 2 articles in feature space. If users tend to edit only one article or the other, you would see 2 clusters of users in feature space. If they edited each article randomly, there would be one random disparate set of users in features space, i.e. the article point locus (2pts) would not look like the user point locus. Since there are many interrelated articles on wikipedia, the locus of points does not form distinct clusters, but it is similar for users and articles.

Conclusion

I have presented a method to detect anomalous Wikipedia edits without knowledge of any content. This is useful because there is an overwhelming amount of edits made to articles everyday and inaccurate edits often go unnoticed. A computationally efficient detection algorithm (i.e. a dot product) would allow edits to be sorted in real time in order of suspicion and shown specifically to field-competent users as determined by their edit history and feedback scores. Such a distributed method would improve the quality of Wikipedia as a whole and make it a more reliable source of information.

References

- Konect Wikipedia Data (http://konect.uni-koblenz.de/networks/edit-enwiki)
- Montanari's EE 378B Lecture Notes (http://www.stanford.edu/class/ee378B/handouts.html)