

Wikipedia Mathematical Models and Reversion Prediction

Jia Ji, Bing Han, Dingyi Li
6/11/2012

Part A. Mathematical Models for Wikipedia Community

I. Introduction

As a free-content encyclopedia project that started 2001, Wikipedia grew quickly over time and had great impact on how people acquire knowledge. According to Alexa's network analysis, Wikipedia ranks 6th among all websites globally in terms of the amount of traffics [1]. The amount of visitors to Wikipedia sites keeps increasing over time, as shown in Figure 1.1.1. However, statistics of site also shows that the amount of editor and edits in the Wikipedia community stops increasing since 2007, as shown in Figure 1.1.2. This seems confusing since one would think the amount of editors should grow proportionally to the number of visitors. Yet the fact indicates that the mathematical model for the amount of editors in Wikipedia community is much more complicated.

In this part of the project we built two models to study the statistics for Wikipedia community. First we developed a predator-prey model which creates a link between Wikipedia communities with a biological ecosystem. The model gives us a more intuitive way to understand the roles that editors and articles play in the community. Then we present a linear regression model that allows us to accurately simulate the evolution progress of the community. Data processing was done by the Elastic-MapReduce on Amazon's server.

This part of the paper is organized as follows, Section II reviews on the related work that has been done before. Section III presents our models for the Wikipedia community. Section IV gives experimental results and makes analysis; Section V makes a conclusion of this part of the project.

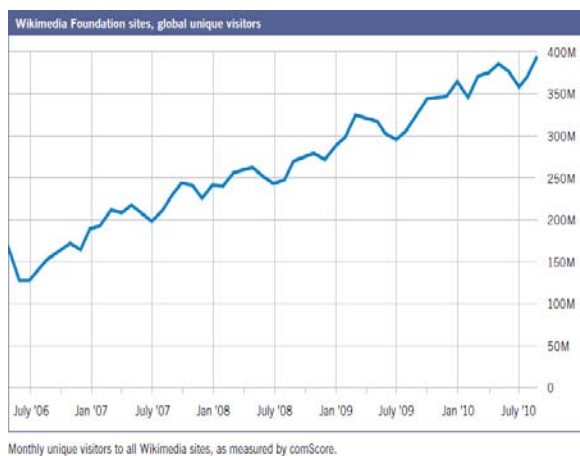


Figure1.1.1 Amount of unique visitors to Wikipedia sites.

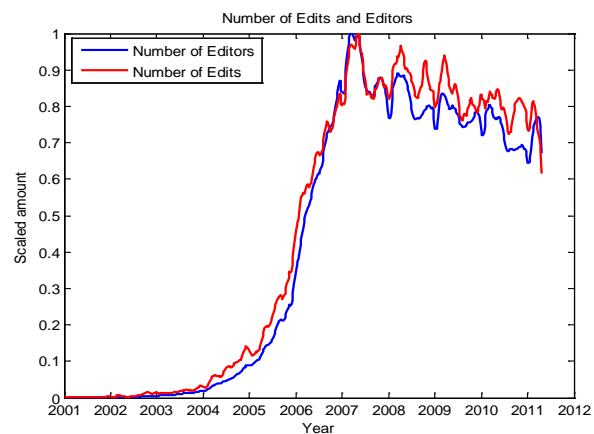


Figure1.1.2 Amount of editors and edits in Wikipedia community

II. Related Work

In the past few decades, numerous Internet-based social media emerge, including email lists, Usenet, MUDs and MOOs, blogs, chat, wikis and social networks. While we research on Wikipedia, other open content online communities can help up obtain many basic understanding of online content

contribution system. There are three basic ideas that have been studied in various online communities [2].

Quantity: Researchers have observed that the minority of participants in online communities provide a majority of the content, which has been found in Usenet postings and in blogging community.

Quality: It has been observed that the quality of Wikipedia's content is mainly determined by the editors. They can simply delete or alter the statement which they find incorrect or disagreeable to them.

Community: The member experience, especially for outsiders, can be greatly influenced by the community norms around the community and diversity within the community.

Based on the research on Wikipedia as well as other online communities, we combine the Wiki community and eco-system together for the first time to examine whether it can give a reasonable explanation of the Wikipedia community.

III. Models

A. Predator-Prey Model

In this model the Wikipedia community is analogous to a biological ecosystem. Editors play the role of predators and articles play the role of preys. When the population of predators and preys are below the capacity of the ecosystem, the amount for both of them would increase exponentially. On the other hand, when the population has reached the capacity, the population would stop increasing and start fluctuating. This type of population growth forms an S-curve. Figure 1.3.1 shows a typical S-curve for biological bio-systems. If we zoom in to look at the fluctuation phase, we would see the interaction between predators and preys. According to the biological ecosystem, when the population of preys increases, there's more food for the predators and as a result the predator's population will increase as well. Then because the predator's population increases, the amount of prey will decrease. As the amount of prey decreases, the predator suffers from shortage of food, and thus their population will also decrease. Therefore fluctuation forms. Figure 1.3.2 shows the fluctuation curve. In the Wikipedia community, we think that an editor feed on articles. When there's many articles to edit, editor's population will grow. But when the editors have edited many articles, they will have trouble with finding an article to edit, and consequently many of them will leave. The decrease of population for editors will in turn cause incoming new articles to pile up, and this will bring back many of the editors that left because of the shortage for articles.

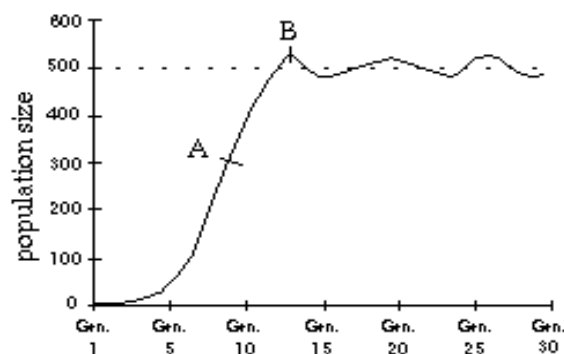


Figure 1.3.1 Population S-curve.

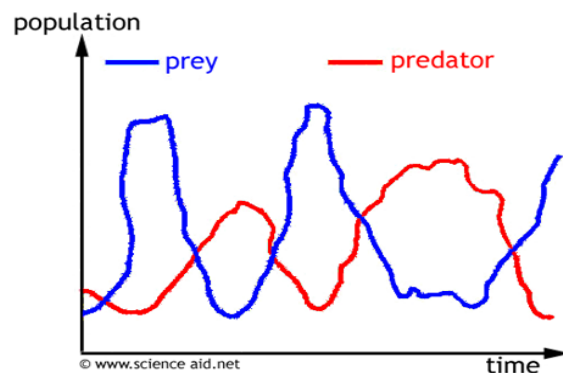


Figure 1.3.2 Predator-prey fluctuation curve.

B. Linear Regression

In the linear regression model, we studied the relationship between the parameters listed in the Table 1.3.1 using linear regression.

Number of Editor	Change in Number of Editor	Number of Active Editor	Number of New Article
Number of Active Article	Total Number of Article	Number of Edit in the Week	Total Number of Edit So Far

Table 1.3.1 Parameters for Linear Regression

We did linear regression for the parameters: “change in the number of editor”, “number of active editor”, “number of new article”, “number of active article”, “number of edit in the week” using other parameters. And in this way, we can also get the model for “number of editor” using “change in the number of editor”, “total number of article” using “number of new article” and “total number of edit so far” from “number of edit in the week”. Here, I will show an example of doing linear regression for the parameter “change in the number of editor”:

Let $F(t)$ denotes the array of parameters at the week t , and $F(t)[i]$ denotes the i th parameter at week t . Do linear regression for $F(2:end)[2]$ using $F(1:end-1)[1,3:8]$ so that we can get the model

$$F(t + 1)[2] = \sum_{i=1,3:8} F(t)[i] * P(i)$$

where $P(i)$'s are the coefficients we got from linear regression.

In this way, we will get the model for each parameter in Table 1.3.1.

IV. Experiments

A. Data

We mainly work on two data sets, DiffDB from Wikipedia and Wikichallenge from Kaggle. DiffDB contains the context difference information between two continuous edits for each article in the entire Wikipedia history [3]. It is as large as around 450GB. The data format for each edit is showed in Table 1.4.1

Wikichallenge is a sampled data set which contains around 44,514 sampled editors and 22,126,031 edits [4]. The data format for each edit is showed in Table 1.4.2.

Rev_Id	Article_Id	Namespace	Title
Timestamp	Comment	Minor	User_Id
User_Text	Diff		

Table 1.4.1 Data Format of DiffDB

User_Id	Article_Id	Revision_Id	Namespace
Timestamp	MD5	Reverted	Reverted_User_Id
Reverted_Rev_Id	Delta	Cur_Size	

Table 1.4.2 Data Format of Wikichallenge

B. Predator-Prey Model

To verify the predator-prey model, we first plotted the population curve for active editors and active articles, shown in Figure 1.4.1. Here active editors are the editors who made at least two edits in two weeks. New articles are the articles that get edited at least twice in two weeks. We can see clear S-shape for the population growth curve, which meet our hypothesis. However we failed to see an obvious lagging between editor's population and article's population. To take a deeper look into the predator-prey relationship between the two, we plotted a cross-correlation function between the first derivatives of their populations, shown in Figure 1.4.2. We can only see a peak at point 0 in this figure. Unfortunately this means there's no lag between population growth of editors and articles. Thus the predator-prey model, although seems right intuitively, is not a right model for Wikipedia community. To analyze the reason why the predator-prey model does not apply to the Wikipedia community, we think of a major difference between editors and predators, and between articles and preys, which is the way they grow the population. Predators and preys, as biological livings, give birth to their next generation directly. Thus the larger their population is, the faster growth rate they have. On the other hand, editors or articles cannot produce more of their kind directly. This would render the whole model based on a wrong assumption, which is that editors are analogous to predators and articles are analogous to preys.

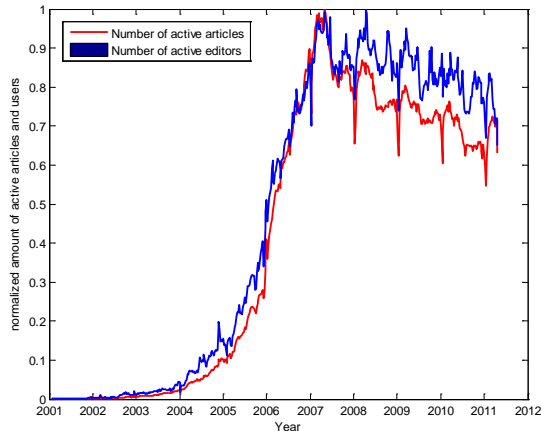


Figure 1.4.1 S-curve for population of active editors and active articles.

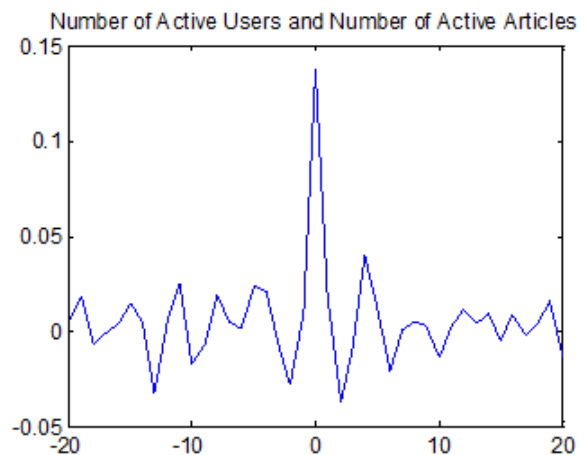


Figure 1.4.2 Cross-correlation function between population of active editors and active articles

C. Linear Regression

To verify the linear regression model, we did a simulation with the model we got. The simulation procedure is as follows:

Let $R(t)$ be the real values of the array of parameters at week t and $S(t)$ be the simulated values of the array of parameters at week t .

1. Set initial value of for the simulation: $S(1) = R(1)$;
2. Choose one parameter i ; give the real values of this parameter: $S(:,i) = R(:,i)$
3. Do simulation of other parameters from the second week to last week: calculate $S(t)[j]$ based on $S(t-1)$ using the models we got in the linear regression step, $j \neq i$

Following above steps, we got the simulation result as shown in figures below:

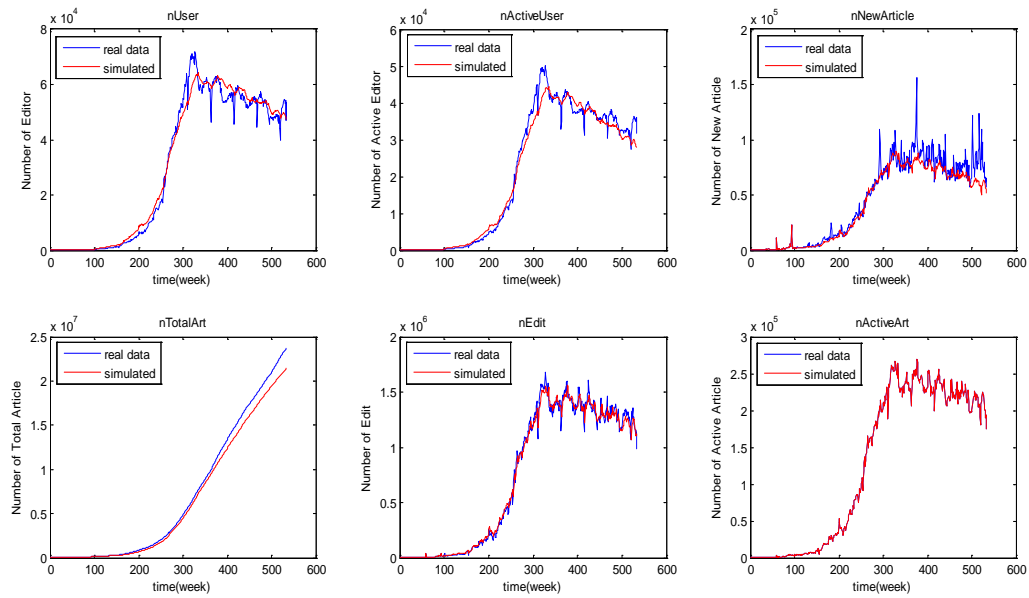


Figure 1.4.3 Simulation Result of Linear Regression Model Given Number of Active Article

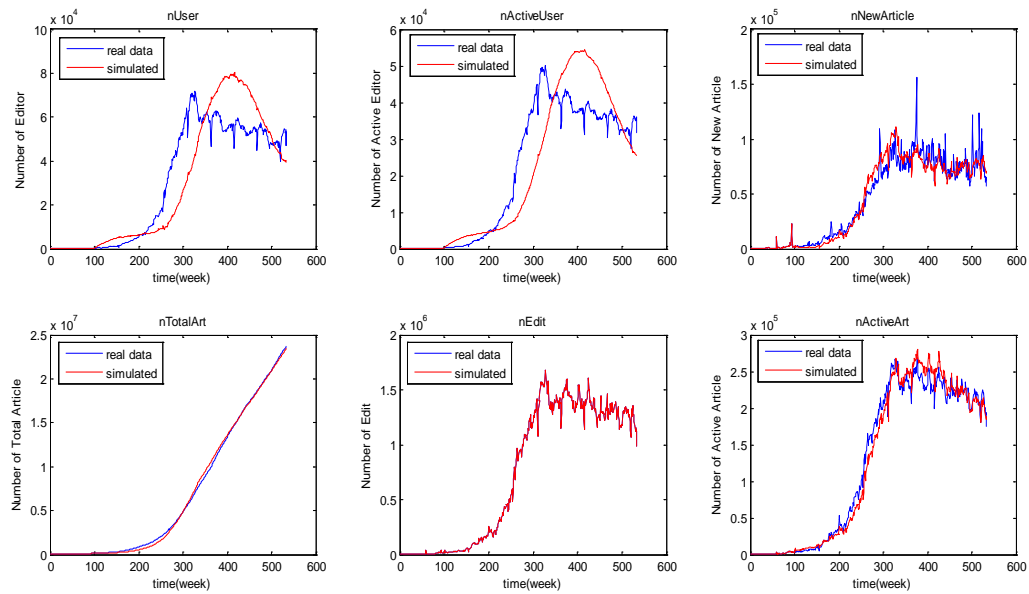


Figure 1.4.4 Simulation Result of Linear Regression Model Given Number of Edit

From the plots, we can find that the given the number of active articles, we can get very good simulation result for all the other parameters; given the number of edits, we can get good result for article parameters, but not for editor parameters; given number of article or number of editor, the simulation result is not as good as given the active article. From that, we can find that the parameters we studied are tightly related to each other and everything has connection.

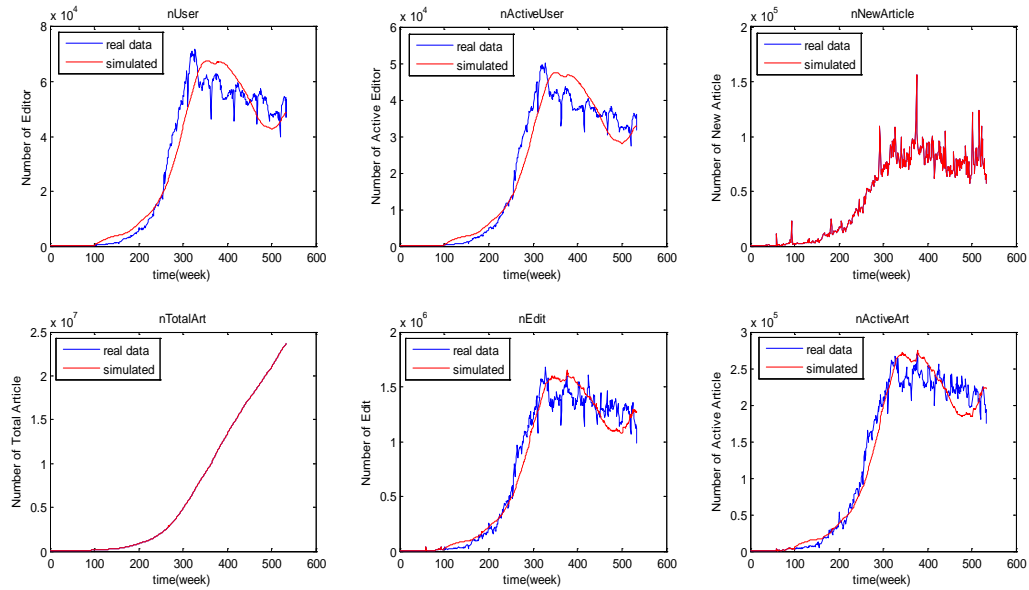


Figure 1.4.5 Simulation Result of Linear Regression Model Given Number of New Article

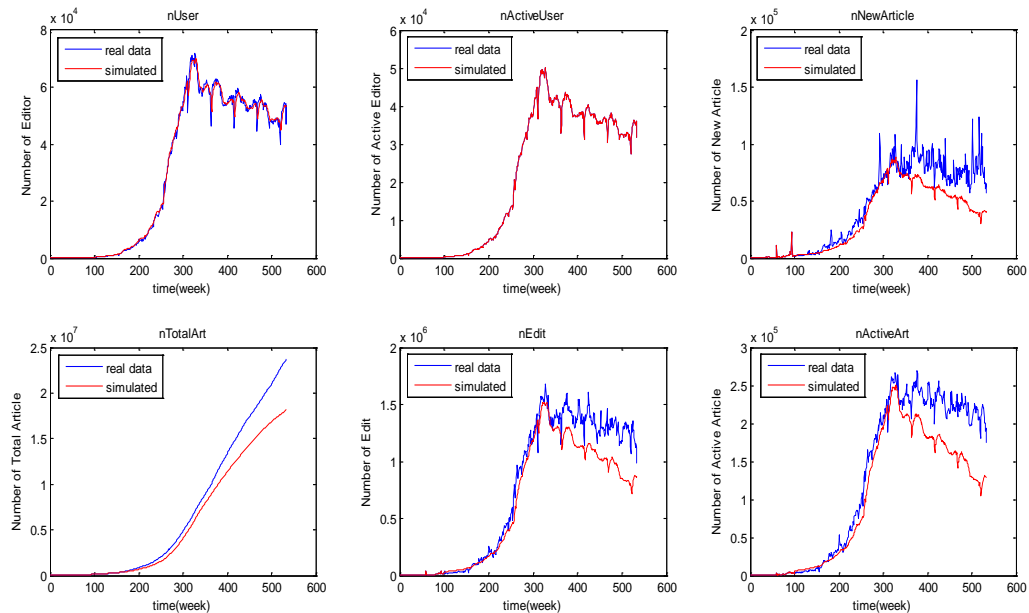


Figure 1.4.6 Simulation Result of Linear Regression Model Given Number of Active Editor

V. Conclusion

In this part of the project, we built two models for the Wikipedia community to explain why the amount of editor stops increasing, whereas the amount of viewers keeps increase. The predator-prey model built a link between the Wikipedia community and the biological ecosystem. This model was proved wrong at the end. We also built a linear regression model, which studies relationship between the many different parameters that describes the Wikipedia community. Given the number of active articles, we get a very nice simulation result for other parameters. This confirms that the linear regression model is a valid model for the Wikipedia community.

Part B. Reversion Prediction

I. Introduction

With the boom of editor and article amount in 2007, we observe that the fraction of reverted edits have a sharp increase as well, shown in Figure 2.1.1. Studying on if an edit will get reverted can lead to very useful outcomes. The result has potential to be used in the areas like controversial topic detection, spammer detection and vandalism detection.

In this part of the project, we utilize three machine learning techniques to predict if an edit will get reverted or not. The techniques we used are Naive Bayes algorithm, logistic regression and SVM. In order to train a model for these algorithms, we extracted three classes of features from an edit, and performed forward feature selection.

This part of the paper is organized as follows, Section II reviews on the related work that has been done before. Section III presents our machine learning procedures to predict reversion. Section IV gives experimental results and makes analysis; Section V makes a conclusion of this part of the project.

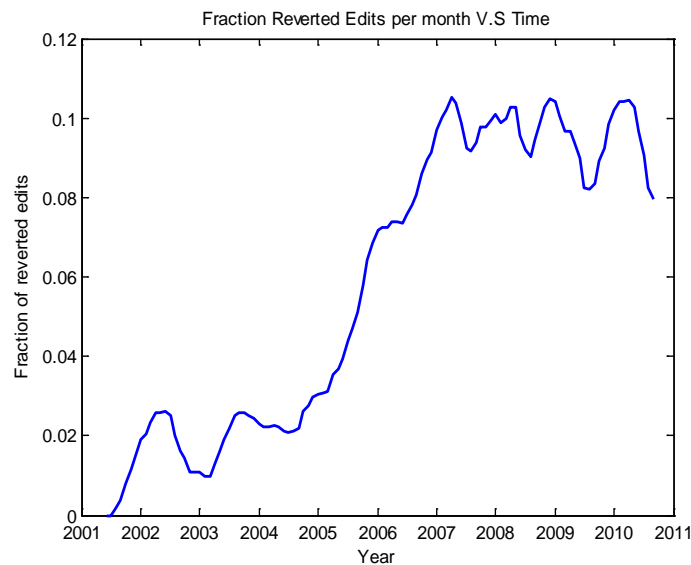


Figure 2.1.1 Fraction of Reverted Edits per Month over time

II. Related Work

There are a variety of reasons for predicting reversion. Many reversion are used to prevent vandalism, a constant problem in an open online encyclopedia. In the past few decades, researchers have developed many methods to identify the potential malicious edits, including examining macro scale feature of edits, vulgarity dictionaries, evaluating text case, and editor characteristics [5]. There are also some researches that make predictions based on the word-level characteristics, which can discover the valuable per-article vandalism behaviors, such as words are vandalistic in one article context but not in another [6]. In this paper, we extract more general features about editor, edit, and article characteristics to train the models and make predictions.

III. Solution

To do reversion prediction, we followed the four steps shown in the workflow in Figure 2.3.1. We first extracted features from the original dataset for each edit; secondly, we did data sampling so that the same number of reverted edits and not reverted edits data were extracted; thirdly, we did machine learning using three machine learning algorithms including Naive Bayes, Logistic Regression and SVM; finally, we did prediction using the model we got in the third step.

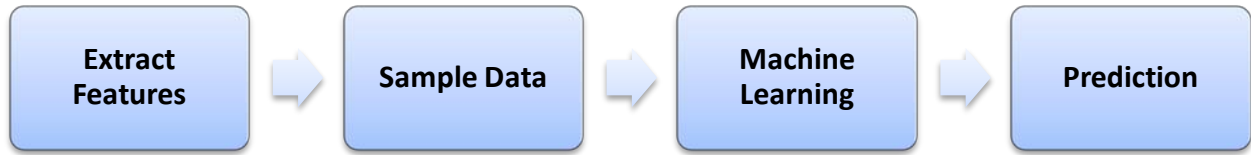


Figure 2.3.1 Workflow of Reversion Prediction

a. Feature Extraction

Feature extraction is the first step of reversion prediction. We extracted three classes of features (24 features in total) for each edit including Article Features, Editor Features, and Edit Features. Details of the features are listed in Appendix A: Feature List for Reversion Prediction.

Map-Reduce Model was used to extract those features from large scale of data. Here, the steps extracting the feature “Editor Age” is shown as example:

Map step: For each edit, emit (editor_id, (edit_time, revision_id)).

Reduce step: Input (editor_id, list(edit_time, revision_id)). First sort the list in ascending order; set start_time as the edit_time of the first (edit_time, revision_id) pair; for each (edit_time, revision_id) pair in the list, we can get the age of the editor when he makes this edit is $age = edit_time - start_time$, then we can emit (revision_id, age).

Combining all of the features we extracted together, we will get an array of feature vectors which include 24 features and a label showing if the edit was reverted or not in each line.

b. Data Sampling

In this step, we sampled data from the feature vectors we got in the first part so that the fraction of reverted edits data and not reverted edits data would both be 50%. Then we divided them into three parts: 50% for training, 20% for validation and 30% for testing.

c. Machine Learning

Machine learning is the key step for reversion prediction. We used three machine learning algorithms to solve the classification problem including Naive Bayes, Logistic Regression and Support Vector Machine (SVM).

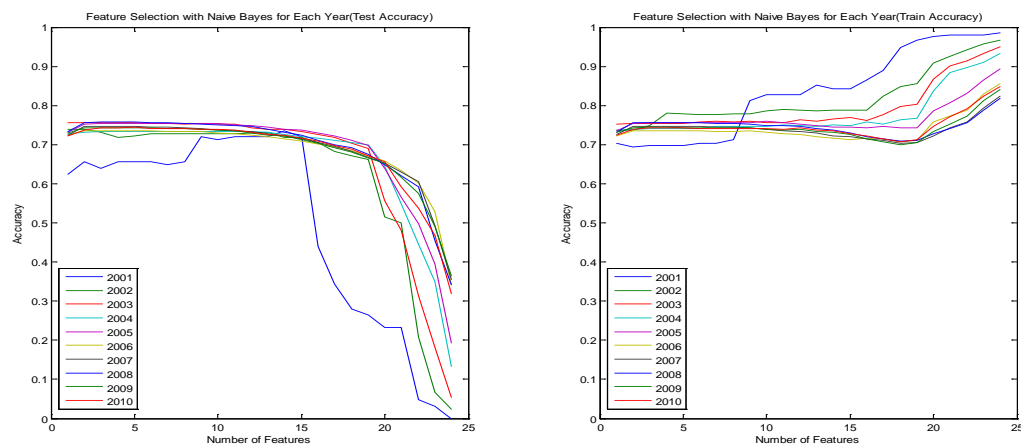


Figure 2.3.2 Feature Selection with Naive Bayes

For each algorithm, we first did a forward feature selection to select the feature set that we want to use. Forward feature selection is a feature selection algorithm that adds one feature at a time that can improve the validation accuracy the most. Figure 2.3.2, Figure 2.3.3, and Figure 2.3.4 shows the feature selection result with Naive Bayes, Logistic Regression and SVM respectively.

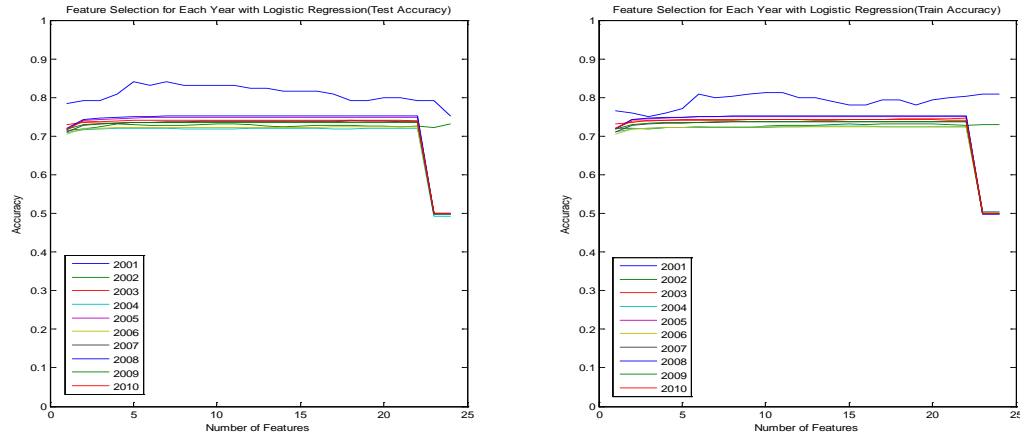


Figure 2.3.3 Feature Selection with Logistic Regression

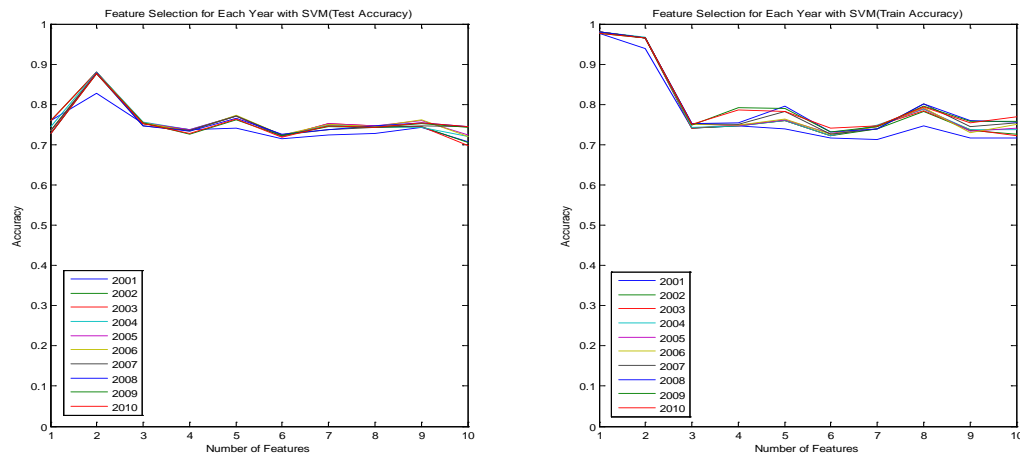


Figure 2.3.4 Feature Selection with SVM

Then, we selected the number of features that can achieve the maximum validation accuracy for each algorithm. In this way, we successfully decided the feature sets that we want to use. Finally, we trained the three models using the training data with the feature sets we selected. In this way, we can do prediction with the models we got.

d. Prediction

In the last step, we can predict if an edit is reverted or not with the models we got in the last step using the features extracted for that edit.

IV. Experiments

As described in the previous part, we trained the models with the selected features and tested on the training set and test set to obtain the train accuracy and test accuracy for each model. Figure 2.4.1

shows the accuracies for all the ten years. All the three models performed similarly and achieved around 75% test accuracy.

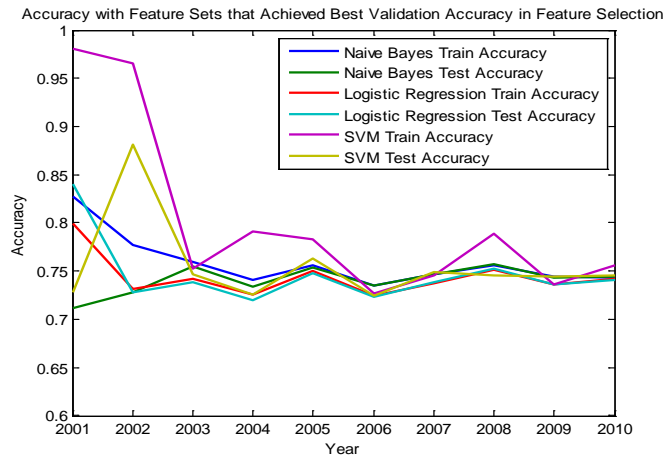


Figure 2.4.1 Prediction Accuracy

Then to test the importance of each class of features, we trained the Naïve Bayes and Logistic Regression model with different feature classes and tested on the training set and test set to obtain the train accuracy and test accuracy. Table 2.4.1 shows the result. We can find that training the model with the three classes separately, Editor Features perform best and Article Features worst. Compared with the performance of Editor Features, combining Editor and Article Features gives similar performance. Compared with the performance of Edit Features, combining Edit and Article Features gives similar performance. Compared with the performance of Edit and Editor Features, combining all three features gives similar performance. These three facts all show that Article Features cannot make a contribution to the prediction if combining with other classes of features. In addition, the combination of Edit and Editor Features give the best prediction performance.

Naive Bayes	Test Accuracy	Logistic Regression	Test Accuracy
Article	0.53355	Article	0.57004
Editor	0.72655	Editor	0.73205
Edit	0.61206	Edit	0.60549
Article+Editor	0.72655	Article+Editor	0.72589
Article+Edit	0.61543	Article+Edit	0.61859
Editor+Edit	0.73729	Editor+Edit	0.73596
All Features	0.73729	All Features	0.73719

Table 2.4.1 Prediction Accuracy of Different Feature Classes with Naive Bayes and Logistic Regression.

To examine the stability of our model, we compare the prediction performance obtained by training with all ten years data and by training with the data of each specific year. The result is shown in Figure 2.4.2. Since the circle and the line matches very well, we can conclude that our model is stable over years. Figure 2.4.3 shows the weight value of each feature in Logistic Regression over ten years. The fact that weights do not change much during the ten years confirms the stability of our models as well.

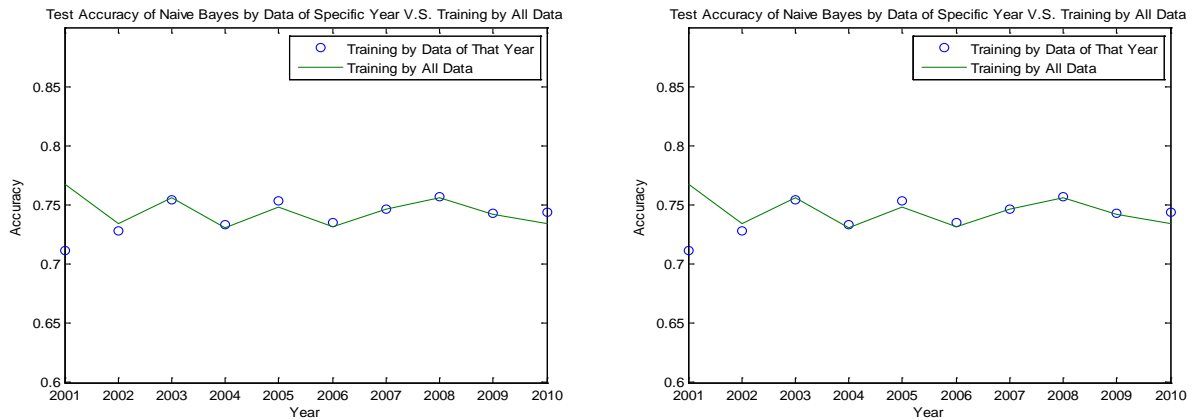


Figure 2.4.2 Training by Data of Specific Year V.S. by All Data with Logistic Regression and Naïve Bayes.

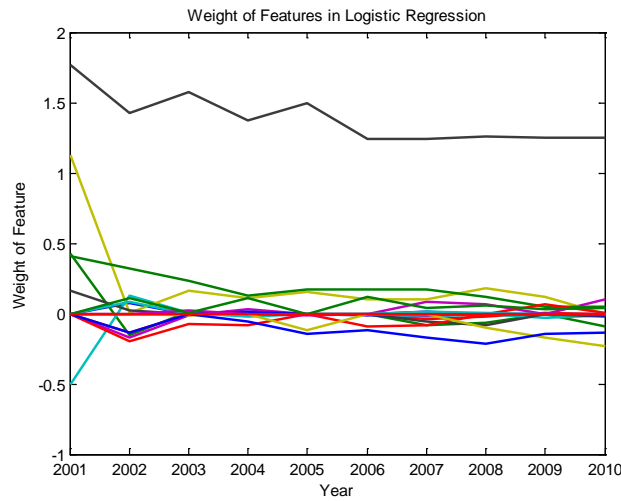


Figure 2.4.3 Weight of Features in Logistic Regression

V. Conclusion

In this part of the project we utilized three machine learning techniques, which are Naive Bayes algorithm, logistic regression and SVM to predict if an edit will be reverted. With the forward selection of features we achieved an accuracy of 75% on a balanced testing dataset, in which contains equal number of reverted and non-reverted edits. Editor feature class contains the most relevant information for the reversion prediction, whereas the article feature class contains the least relevant information. We also studied the stability of our model, and came to a conclusion that the model for reversion prediction hasn't changed since Wikipedia was founded in 2001.

Reference

- [1] Alexa.com <http://www.alex.com/siteinfo/wikipedia.org>
- [2] Katherine Panciera, Aaron Halfaker, Loren Terveen. Wikipedians are born, not made: a study of power editors on Wikipedia. GROUP '09 Proceedings of the ACM 2009 international conference on Supporting group work (2009), ACM Press, NY, USA, pp.51-60.
- [3] http://meta.wikimedia.org/wiki/WSoR_datasets/revision_diff
- [4] <http://www.kaggle.com/c/wikichallenge/data>
- [5] Potthast, M., Stein, B., and Gerling, R. Automatic vandalism detection in Wikipedia. Advances in Information Retrieval, 2008, 663–668.
- [6] Jeffrey Rzeszotarski, Aniket Kittur. Learning from history: predicting reverted work at the word level in Wikipedia. CSCW '12 Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work. (2012) ACM Press, NY, USA, pp. 437-440.

Appendix A: Feature List for Reversion Prediction

1	ArtAge	Age of the article	13	LogEditorNumArt	Log(Number of article the editor has edited)
2	ArtEditNum	Number of edits made on the article	14	EditorRevertFrac	Fraction of reverted edits made by the editor
3	ArtIsNew	If the article has not been edited for more than 6 months	15	LogEditorRevertFrac	Log(Fraction of reverted edits made by the editor)
4	ArtIsActive	If the article has been edited more than 10 times within one month	16	NameSpace	Namespace of the article
5	ArtNumEditor	Number of editor who has edited the article	17	Delta	Change of characters compared with the previous revision of the article
6	ArtRevertFrac	Fraction of reverted edits made on the article	18	SignDelta	Increase / Decrease number of characters
7	EditorAge	Age of the editor	19	AbsDelta	Length of the edit
8	EditorEditNum	Number of edit made by the editor	20	Curr_Size	Current size of the article
9	LogEditorEditNum	Log(Number of edit made by the editor)	21	ChangeRate	AbsDelta / Curr_Size
10	EditorIsNew	If the editor has not edited for more than 6 months	22	LogDelta	Log(Change of characters compared with the previous revision of the article)
11	EditorIsActive	If the editor has edited more than 10 times within one month	23	LogAbsDelta	Log(Length of the edit)
12	EditorNumArt	Number of article the editor has edited	24	LogChangeRate	Log(AbsDelta / Curr_Size)