Bauhaus-Universität Weimar
Faculty of Media
Degree Programme Computer Science and Media

Advanced Vandalism Detection on Wikipedia

Master's Thesis

Paul Christoph Götze
Born Oct. 17, 1988 in Dessau

Matriculation Number 110818

1. Referee: Prof. Dr. Benno Stein
2. Referee: Prof. Dr. Charles A. Wüthrich

Submission date: September 3, 2014

Declaration

Unless otherwise indicated in the text or references, this thesis is entirely the product of my own scholarly work.

Weimar, September 3, 2014

. . . .
Paul Christoph Götze

Abstract

This thesis shows alternative approaches for automatic vandalism detection on the English Wikipedia online encyclopedia. Vandalism on Wikipedia is defined as offensive edits that attempt to compromise the integrity of the encyclopedia. In order to prevent vandalism, machine learning approaches (classifiers) can be used to predict whether a contribution is a vandalism edit or a regular edit.

Learning classification models on imbalanced datasets, such as the PAN-WVC corpora, is often reported to result in low classification performance. Based on our own implementation of a Wikipedia vandalism detection system, we examine the impact of resampling the training dataset on classification performance. We compare four different classifiers, which are learned on datasets resampled by random undersampling, SMOTE, and a combination of both.

As a second approach to overcome the class imbalance problem, we consider the Wikipedia vandalism detection task as one-class classification problem. We evaluate the one-class classifier by Hempstalk et al. [2008] and the one-class SVM as implemented by Chang and Lin [2011] on the PAN-WVC corpora.

In order to create unsupervised vandalism detection approaches, we use the full revision history of Wikipedia articles to compile simple vandalism datasets and evaluate the resulting article-relative classifiers. In a pilot study, we introduce the idea of category-relative classifiers and investigate the similarity of vandalized contents between different articles, as well as article categories. Attempts are made to combine several article-relative classifiers in order to reach a higher classification performance.

Contents

Chapter 1

Introduction

This thesis deals with alternative classification approaches for automatic vandalism detection on Wikipeida. Wikipedia is the largest online open-content collaborative encyclopedia with about 32.2 million articles in total[1] (June 2014) considering all languages. The most popular Wikipedia is the English one with about 4.6 million articles. Wikipedia articles are open to be edited by anyone anonymously. This leads to several forms of nonconstructive contributions—so called vandalism. Vandalism is defined by Wikipedia itself as:

> [...] *any addition, removal, or change of content in a deliberate attempt to compromise the integrity of Wikipedia.*[2]

Acts of vandalism can be seen as violations of one or more rules given by Wikipedia's five pillars, which shall help to ensure the mentioned *integrity*:[3]

1. *Wikipedia is an encyclopedia.* It should not be used for advertising, journalism or original research content.

2. *Wikipedia is written from a neutral point of view.* Personal opinions and experiences have to be neglected and multiple viewpoints held by the majority should be available.

3. *Wikipedia is free content that anyone can edit, use, modify, and distribute.* A User account is not needed to make contributions.

4. *Editors should treat each other with respect and civility.* Users are not allowed to use vulgarism and abuse.

[1]`http://stats.wikimedia.org/EN/TablesArticlesTotal.htm`
[2]`https://en.wikipedia.org/wiki/Wikipedia:Vandalism`
[3]`https://en.wikipedia.org/wiki/Wikipedia:Five_pillars`

5. *Wikipedia does not have firm rules.* There are no exact ways of editing an article. Thus, mistakes made by new users are common due to missing knowledge.

Several studies showed the importance of preventing vandalism in Wikipedia articles. The estimated percentage of vandalism edits varies from 1–2% (Kittur et al. [2007], Tran and Christen [2013]) to around 5% in a Wikipedia-internal study,[4] up to 7% (Potthast [2010]). The mean survival time of non-constructive contributions has been found to be 2.1 days, with a median time of 11.3 minutes (Kittur et al. [2007]). This shows that users spend a lot of time on identifying and reverting vandalism to prior revisions. It has been shown by Geiger and Halfaker [2013], that the absence of non-human patrols (so-called *bots*) dramatically increase the survival time of vandalized contents in Wikipedia articles. Thus, automatic vandalism detection systems help to facilitate the effort of a manual detection and enable users to focus more on writing contributions to the encyclopedia instead of doing time-consuming maintenance tasks.

1.1 Research Questions

This thesis questions existing approaches for Wikipedia vandalism detection. It tries to find alternative classification approaches to solve the vandalism detection challenge by a systematic study of the class imbalance problem of vandalism edits versus regular edits and by treating vandalism detection as a one-class classification problem. We contribute by answering the following research questions:

1. **How does resampling of a skewed training dataset influence Wikipedia vandalism detection performance?**

   Since resampling a skewed training dataset has often been found to be advantageous to classification performance (for several surveys see Ganganwar [2012], Guo et al. [2008], He and Garcia [2009], Kotsiantis et al. [2006]), we examine the impact of balancing and SMOTE oversampling the training dataset on the detection performance of different classifiers. We found training dataset resampling to improve the classification performance for three of four classifiers. However, a Random Forest classifier without training data resampling still achieves the highest overall performance.

[4] `https://en.wikipedia.org/wiki/Wikipedia:Counter-Vandalism_Unit/Vandalism_studies/Study1`

2. **What is the performance of a one-class classifier in detecting vandalism on Wikipedia relative to binary classifiers used so far?**

   The most salient property of a one-class classifier is that it uses only the target class (here the samples tagged as "vandalism") from the training dataset to train the classifier.

   Since the Wikipedia vandalism detection task can be seen as a one-class classification problem (see Section 3.2), the one-class classifier of Hempstalk et al. [2008] and one-class SVM as implemented by Chang and Lin [2011] is used for detecting vandalism. We found that both one-class classifiers perform worse than a Random Forest classifier on commonly used Wikipedia vandalism detection corpora. We attempt to explain why this is the case and what steps can be used to improve the classification performance.

3. **Is there an advantage in using a classifier for each article or category? How can an unsupervised or semi-supervised classifier for each article or category be created?**

   In a pilot study, we investigate the creation of unsupervised classifiers that are learned on training datasets compiled from the revision histories of 12 English Wikipedia articles. Considering reverted edits in the articles' revision histories as simple vandalism cases, we examine the feasibility of category-relative classifiers. Evaluating the similarity of vandalism contents in the article-relative datasets and article categories, our study does not provide evidence for the applicability of category-relative vandalism detection.

   Furthermore, we evaluate the 12 article-relative classifiers on one of the publicly available Wikipedia vandalism test datasets. By combining the article-relative datasets and classifiers, we improve the classification performance regarding the single classifiers learned on the article-relative datasets. Using the simple vandalism approach, we found that enriching a commonly used Wikipedia vandalism corpus with vandalism samples compiled from the article-relative datasets does not improve the classification performance on the mentioned vandalism corpus.

1.2 Organization of the Thesis

Chapter 2 introduces in formal terms the terminology of classification tasks in general, and the vandalism detection task in particular. Existing approaches for vandalism detection on Wikipedia are reviewed and compared to each other with respect to their performances obtained from earlier evaluations. Finally, our developed Wikipedia vandalism detection system is described and evaluated. Chapter 3 considers techniques to enhance detection performance. We examine the suitability of training dataset resampling and one-class classifiers for vandalism detection on Wikipedia. In Chapter 4, edits compiled from the history dump files of the English Wikipedia are classified. Findings are discussed which motivate the development of alternative classifiers based on a single article and category. Finally, Chapter 5 summarizes the contributions of this thesis and gives some directions for future work.

Chapter 2

Wikipedia Vandalism Detection

In this chapter we formally introduce Wikipedia vandalism detection as a classification task. Moreover, the performance measures that are used to evaluate a classifier's performance are discussed. These measures are employed within our experiments in Chapter 3 and Chapter 4. Furthermore, we provide an overview of existing vandalism detection approaches based on machine learning as well as corpora that are used for training and evaluation. We compile a comprehensive list of features which were used in the literature so far. A selection of these features are employed in our own vandalism detection system described in the subsequent section. The chapter closes with an evaluation of our system in comparison to the best performing approaches from the literature.

2.1 Problem Definition

Using machine learning, nonconstructive contributions can be detected by inspecting Wikipedia edits. An edit $e = (r_{old}, r_{new})$ is defined as a set of two consecutive revisions of an article comprising the original revision r_{old} and the new revision r_{new} after the changes are submitted to Wikipedia. A revision r is a version of a Wikipedia article that, besides the article markup text, includes meta data about the latest editing, such as the editor's user identification, her comment on the nature of the changes made, and a timestamp at which she edited the article.

To determine whether an edit e is vandalism, classification algorithms (classifiers) are applied. These classifiers estimate the class membership of an edit based on numerical features which are calculated from the edit's old and new revision. Let $\mathcal{F} = \{f_1, f_2, \ldots\}$ denote such a set of features where each feature f_i is a function mapping an edit onto a real number $f_i : e \to R$. Thus, an edit can be represented by its feature vector $\mathbf{e} = \mathcal{F}(e) = \{f_1(e), f_2(e), \ldots\}$.

Let $\mathbf{F} = \{\mathbf{e_1}, \mathbf{e_2}, \ldots\}$ denote a set of feature vectors of the edits $(e_1, e_2, \ldots)$. Given a vandalism corpus $\mathcal{C} = \{(e_1, 0), (e_2, 1), \ldots\}$ comprising a representative sample of vandalism edits $(e_i, 1)$ and possibly, but not necessarily regular edits $(e_i, 0)$, a linear classifier $c, c : \mathbf{F} \to \{0, 1\}$ can be trained, so that an unlabeled edit e can be classified by computing $c(e)$, which maps the edit's feature vector $\mathbf{e}$ onto 0 (regular edit) or 1 (vandalism edit).

2.2 Corpora

To ensure the comparability of classification approaches on Wikipedia vandalism detection, several evaluation corpora were compiled and are commonly used in the literature. This section outlines the most important ones as well as the Wikipedia history dump corpus used in our research.

Webis-WVC-07 The Webis Wikipedia Vandalism Corpus 2007 (Webis-WVC-07)[1] was the first Wikipedia vandalism corpus and consists of 940 human-annotated edits of which 301 are labeled as vandalism. It was compiled in 2007 and was first used by Potthast et al. [2008]. All edits were taken from the English Wikipedia and were chosen from articles with high conflict potential and by using the results of a Wikipedia-internal vandalism study.

PAN-WVC-10 The PAN Wikipedia Vandalism Corpus 2010 (PAN-WVC-10), compiled in 2010 via Amazon's Mechanical Turk,[2] comprises 32439 edits from 28468 English Wikipedia articles of which 2394 have been annotated as vandalism. 753 human annotators created the resulting dataset casting 193022 votes, so that each edit has been annotated at least three times, whereas edits that were difficult to be annotated received more than three votes (Potthast [2010]). The training dataset provides the annotated edits' revision texts and meta data (id, comment, timestamp), the old revisions' id, as well as the article name and a URL to the respective diff-page on Wikipedia. Additionally, it comprises meta data of the ground truth annotation process, such as the annotator's id, annotation time, and submission time. Whereas the editors' name or ip is missing in the training dataset, the test dataset additionally provides the editor's name and the article id. The PAN-WVC-10 was first used in the 1st International Competition on Wikipedia Vandalism Detection (Potthast et al. [2010]).

[1] The Webis-WVC and PAN-WVC corpora are available on `http://www.webis.de/research/corpora/`.

[2] `http://www.mturk.com/`

PAN-WVC-11 The PAN Wikipedia Vandalism Corpus 2011 (PAN-WVC-11) from 2011 is an extension of the PAN-WVC-10. It was used in the 2nd International Competition on Wikipedia Vandalism Detection (Potthast and Holfeld [2011]) and is the first multilingual vandalism detection corpus. The corpus comprises 29949 Wikipedia edits in total (9985 English edits with 1144 vandalism, 9990 German edits with 589 vandalism, and 9974 Spanish edits with 1081 vandalism annotations). As training dataset, all edits from the PAN-WVC-10 training and test corpora are used. Both the training and test corpora of PAN-WVC-11 provide the annotated edits' revision texts, the new revision's meta data (id, user name, comment, timestamp), the old revision's id, as well as the article name and id, and a URL to the Wikipedia diff-page. Furthermore, similar to PAN-WVC-10, annotation meta data, such as the number of annotators per edit, is provided.

Wikipedia History Dump Wikipedia records all revisions of all articles and all other Wikipedia pages and releases them as XML or SQL dump files.[3] For the experiments conducted in this thesis we used the XML pages meta history dump from June 14th, 2014. In the dump files named `pages-meta-history`, all articles' revision histories, including markup text and revision meta data, are provided. Dump files named `stub-meta-history` solely comprise the full revision history with meta data, but do not include markup texts.

2.3 Performance Measures

To evaluate the performance of a trained classifier in relation to a test dataset, the detection performance measures *precision* and *recall* are used. These performance measures are computed from error values counted in a classifier's confusion matrix:

Classifier	**Actual**	
Prediction	P	N
P	TP	FP
N	FN	TN

In our case P (positive) denotes vandalism edits and N (negative) denotes regular edits. Hence, TP counts true positive detections, FP count false positive detections, FN denotes false negative detections, and TN counts true negative detections. From the values given in the confusion matrix the prediction error measures precision and recall can be computed as follows:

[3]`http://dumps.wikimedia.org/enwiki/`

$$\text{precision} = \frac{TP}{TP + FP} \tag{2.1}$$

$$\text{recall} = \frac{TP}{TP + FN} \tag{2.2}$$

Most classifiers return a confidence value instead of a class label for a tested data sample. The final classification has to be done by choosing a threshold value that determines the class by being compared with the obtained confidence. By varying the threshold for the vandalism confidence values, the prediction error measures can be computed for each possible threshold. Thus, to create the precision-recall (PR) curve, the sorted recall values are plotted against the precision values varying the classification threshold. Analyzing the area under the curve avoids the problem of choosing a threshold a priori. Furthermore, an evaluation of the area under the curve allows for making a statement about the suitability of a certain classifier for certain detection situations, and whether a threshold can be chosen to meet detection criteria. Besides precision and recall, the false-positive rate (FP-rate) and the true-positive rate (TP-rate) can be computed as follows:

$$\text{FP-rate} = \frac{FP}{FP + TN} \tag{2.3}$$

$$\text{TP-rate} = \frac{TP}{TP + FN} \tag{2.4}$$

Note that the TP-rate is equal to recall. Adler et al. [2010], Harpalani et al. [2011] and Javanmardi et al. [2011] solely used the area under receiver-operator characteristic curve (ROC-AUC) which plots the FP-rate values against the TP-rate values varying the classification threshold. Most recent work as well as the PAN competitions also used the area under precision-recall curve (PR-AUC). Since the curve in receiver-operator characteristic (ROC) space only dominates if and only if it dominates in PR space (Davis and Goadrich [2006]), we also choose the PR-AUC as the more significant measure for the classifier's performance.

2.4 Related Work

The following sections present early community-initiated techniques to detect vandalism on Wikipedia and provide an overview of machine learning-based approaches from the literature.

2.4.1 Wikipedia Bots

The vandalism problem on Wikipedia is as old as the encyclopedia itself. Kittur et al. [2007] observe that the total number of vandalism edits is increasing over time. Although they report the total vandalism proportion to remain at the same level, increasing vandalism is a serious objective in the online encyclopedia. To tackle this problem, the Wikipedia community resorts to manually protecting articles from being edited in case they are heavily vandalized. Additionally, since 2006, vandalism detection bots are used, which automatically patrol for vandalism edits and partly revert them. Most often these bots use simple heuristic rules, word blacklists, and lists of blocked user IPs to identify vandalism edits (e.g. VoABot II[4] or ClueBot[5]). The ClueBot NG bot,[6] which replaces ClueBot, uses machine learning approaches. Thus, it tries to enhance the heuristics-based techniques, which appeared to be difficult to maintain and easy to bypass. The bot uses a preclassified edit dataset annotated by Wikipedia users to train an Artificial Neural Network. The classifier operates on calculated edit features, such as different word-level vandalism probabilities, to classify new edits.[7]

2.4.2 Machine Learning Approaches

Since 2008 Wikipedia vandalism detection based on machine learning approaches has become a field of increasing research interest. Table 2.1 outlines the existing vandalism detection approaches from the literature, including information about which corpora and classifiers were used for training, as well as the obtained detection performances.

Potthast et al. [2008] contributed the first machine learning vandalism detection approach using textual features as well as basic meta data features with a logistic regression classifier. Smets et al. [2008] used a Naive Bayes classifier on a bag of words edit representation and were the first to use compression models to detect Wikipedia vandalism. Itakura and Clarke [2009] used Dynamic Markov Compression to detect vandalism edits on Wikipedia. Mola Velasco [2010] extended the approach of Potthast et al. [2008] by adding some additional textual features and multiple wordlist-based features. He was the winner of the *1st International Competition on Wikipedia Vandalism Detection* (Potthast et al. [2010]). West et al. [2010] were among the first to present a vandalism detection approach solely based on spatial and temporal meta data,

[4]`http://en.wikipedia.org/wiki/User:VoABot_II`

[5]`http://en.wikipedia.org/wiki/User:ClueBot`

[6]`http://en.wikipedia.org/wiki/User:ClueBot_NG`

[7]Unfortunately, up to now (Aug. 2014), the dataset used by ClueBot NG is not publicly accessible or available to download.

Table 2.1: Vandalism detection classification results from the literature.

Author	Corpus	Balanced	Classifier	Precision	Recall	PR-AUC
Smets et al. [2008]	6944 labeled by revert comment	no	Probab. Sequence Modeling	.320	.920	–
Smets et al. [2008]	6944 labeled by revert comment	no	Naive Bayes	.410	.570	–
Tran and Christen [2013]	history dump/article view count	yes	Gradient Tree Boosting	.870	.870	–
Potthast et al. [2008]	Webis-WVC-07	no	Logistic Regression	.830	.770	–
Mola Velasco [2010]	PAN-WVC-10	no	Random Forest	.860	.570	.660
Wang and McKeown [2010]	PAN-WVC-10	yes	LogitBoost	.850	.860	–
Adler et al. [2010]	PAN-WVC-10	no	ADTree	.370	.770	.490
Adler et al. [2011]	PAN-WVC-10	no	Random Forest	–	–	.820
West and Lee [2011]	PAN-WVC-10	no	ADTree	–	–	.750
Harpalani et al. [2011]	PAN-WVC-10	no	LogitBoost	.735	.477	–
Our current system	PAN-WVC-10	no	Random Forest	.606	.608	.671
West and Lee [2011]	PAN-WVC-11	no	ADTree	–	–	.820
Our current system	PAN-WVC-11	no	Random Forest	.927	.399	.748

without the need to inspect article or revision texts. Similarly Adler et al. [2010] built a vandalism detection system on top of their WikiTrust reputation system (Adler and De Alfaro [2007]). Adler et al. [2011] combined natural language, spatial, temporal and reputation features used in their aforementioned works (Adler et al. [2010], Mola Velasco [2010], West et al. [2010]). Besides Adler et al. [2011], West and Lee [2011] were the first to introduce ex post facto data as features, for whose calculation also future revisions have to be considered. Their resulting multilingual vandalism detection system was the winner at the *2nd International Competition on Wikipedia Vandalism Detection* (Potthast and Holfeld [2011]). Harpalani et al. [2011] stated vandalism edits to share unique linguistic properties. Thus, they based their vandalism detection system on a stylometric analysis of vandalism edits by probabilistic context-free grammar models. They showed that this approach outperforms features based on shallow patterns, which match syntactic structures and text tokens. Supporting the current trend of creating cross language vandalism classifiers, Tran and Christen [2013] evaluated multiple classifiers based on a set of language independent features that were compiled from the hourly article view counts and Wikipedia's complete edit history.

2.4.3 Vandalism Features

The literature provides a growing set of features that are employed to model vandalism edits. After the first contributions to the Wikipedia vandalism detection task, most authors used a subset of existing features and added some new ones to their approaches. Table 2.2 and Table 2.3 provide a comprehensive overview of textual and meta data vandalism features that were used so far in the literature. The following abbreviations are used to refer to authors: P08 (Potthast et al. [2008]), M10 (Mola Velasco [2010]), Wa10 (Wang and McKeown [2010]), We10 (West et al. [2010]), A10 (Adler et al. [2010]), A11 (Adler et al. [2011]), W11 (West and Lee [2011]) and J11 (Javanmardi et al. [2011]).

Textual features are calculated by analyzing the new revision's markup text or rather both revsions's markup texts of an edit. Meta data features are compiled from the revision's meta data or are calculated by analyzing additional Wikipedia data, such as history dumps or article dumps. While Mola Velasco [2010] used three feature categories by considering *textual*, *meta data* and *language features*, we categorized his language features (wordlist-based features) to be textual features. Javanmardi et al. [2011] split their features into four categories, namely *textual*, *meta data*, *user* and *language model*. The *user* category comprises user-related meta data features. Language model features are Kullback-Leibler Divergence-related features which are calculated based on the markup text. Thus, features belonging to their *user* category can be found in our meta data features and *language model* features are arranged under textual features.

2.5 Our Vandalism Detection System

Based on existing vandalism detection systems, we developed our own Wikipedia vandalism detection system[8] as a JRuby[9] extension package (gem). Our system is based on several features from Adler et al. [2011], Javanmardi et al. [2011], Mola Velasco [2010], Potthast et al. [2008], Wang and McKeown [2010] and West and Lee [2011] with four additionally added or adjusted features. Therefore, we rated the features given in Section 2.4.3 according to their implementation effort and calculation expense. Furthermore, we did not consider features that use ex post facto data or need to extensively request the Wikipedia API or to process Wikipedia dump files. Table 2.4 and Table 2.5 organize

[8]Our Wikipedia vandalism detection system is available on `https://github.com/webis-de/wikipedia-vandalism-detection`

[9]`http://jruby.org/`

Table 2.2: Textual features to describe vandalism edits from the literature.

Feature	P08	M10	Wa10	We10	A10	A11	W11	J11	Our
LONGEST CHAR SEQUENCE	✓	✓				✓	✓	✓	✓
COMPRESSIBILITY	✓	✓				✓		✓	✓
UPPER TO ALL RATIO	✓	✓				✓	✓	✓	✓
AVG. TERM FREQUENCY	✓	✓				✓			✓
LONGEST WORD	✓	✓				✓	✓	✓	✓
PRONOUN FREQUENCY	✓	✓				✓	✓	✓	✓
PRONOUN IMPACT	✓	✓				✓	✓	✓	✓
VULGARISM FREQUENCY	✓	✓	✓				✓	✓	✓
VULGARISM IMPACT	✓	✓					✓	✓	✓
SIZE RATIO	✓	✓	✓			✓		✓	✓
REPLACEMENT SIMILARITY	✓								✓
CONTEXT RELATION	✓								
UPPER TO LOWER RATIO		✓				✓		✓	✓
DIGIT RATIO		✓				✓		✓	✓
NON-ALPHANUMERIC RATIO		✓						✓	✓
ALPHANUMERIC RATIO						✓	✓		
CHARACTER DIVERSITY		✓						✓	✓
SIZE INCREMENT		✓				✓	✓		✓
BIASED WORDS FREQUENCY		✓				✓		✓	✓
SEX WORDS FREQUENCY		✓				✓		✓	✓
BAD WORDS FREQUENCY		✓				✓		✓	✓
GOOD/MARKUP WORDS FREQUENCY		✓				✓		✓	✓
ALL WORDS FREQUENCY		✓				✓		✓	✓
BIASED WORDS IMPACT		✓				✓		✓	✓
SEX WORDS IMPACT		✓				✓		✓	✓
BAD WORDS IMPACT		✓				✓		✓	✓
GOOD/MARKUP WORDS IMPACT		✓				✓		✓	✓
ALL WORDS IMPACT		✓				✓		✓	✓
WEB SLANG FREQUENCY			✓						
EMOTICONS FREQUENCY									✓
EMOTICONS IMPACT									✓
PUNCTUATION MISUSE			✓						
ARTICLE SIZE							✓		✓
CHARACTERS ADDED OR REMOVED							✓		✓
MODIFIED NON-ADJACENT TEXT BLOCKS							✓		
INSERTED WORDS								✓	
REMOVED WORDS								✓	
WORDS INCREMENT									✓
BLANKING								✓	✓
INTERNAL LINKS ADDED								✓	✓
EXTERNAL LINKS ADDED								✓	✓
INSERTED WIKI MARKUP							✓		
REMOVED WIKI MARKUP							✓		
KL-DIVERGENCE, INSERTED (CHAR DISTR.)	✓	✓				✓		✓	✓
KL-DIVERGENCE, REVISIONS								✓	✓
KL-DIVERGENCE, REMOVED								✓	✓

Table 2.3: Meta data features to describe vandalism edits from the literature. Features annotated with an asterisk $*$ need Wikipedia API requests while using the PAN-WVC training and tests corpora. Features annotated with a plus sign $+$ use ex post facto data (future revisions).

Feature	**P08**	**M10**	**Wa10**	**We10**	**A10**	**A11**	**W11**	**J11**	**Our**
ANONYMITY	✓	✓			✓	✓	✓		✓
COMMENT LENGTH	✓	✓		✓	✓	✓	✓	✓	✓
EDITS PER USER*	✓		✓						✓
COMMENT CUE WORDS FREQUENCY			✓						✓
TIME-OF-DAY				✓	✓	✓	✓		✓
WEEKDAY				✓		✓	✓		✓
TIME SINCE REGISTRATION*				✓		✓			
TIME SINCE LAST ARTICLE EDIT*				✓		✓	✓	✓	
TIME SINCE LAST OFFENDING EDIT*				✓		✓	✓		
REGISTERED USER'S PROPERTIES*				✓				✓	
ARTICLE REPUTATION*				✓		✓	✓		
USER REPUTATION*				✓	✓	✓	✓	✓	
CATEGORY REPUTATION*				✓		✓			
COUNTRY REPUTATION*				✓		✓	✓		
TIME TO PREV. REVISION*					✓	✓	✓		✓
TIME TO NEXT REVISION$^+$					✓	✓			
MIN. REVISION QUALITY					✓	✓			
TOTAL WEIGHT OF JUDGES*					✓	✓			
AVG. REVISION QUALITY					✓	✓			
MAXIMUM DISSENT					✓	✓			
DELTA (EDIT DISTANCE)					✓				
NEXT REVISION COMMENT LENGTH$^+$					✓	✓			
NEXT COMMENT MENTIONED A REVERT$^+$					✓	✓			
PREV. TEXT TRUST HISTOGRAM*					✓	✓			
CURRENT TEXT TRUST HISTOGRAM*					✓	✓			
TRUST HISTORGRAM DIFFERENCE*					✓	✓			
EDITOR SAME AS PREVIOUS*						✓	✓		✓
PREV. REVISION LENGTH						✓			
NEXT EDITOR ANONYMOUS$^+$						✓			
JUDGES NUMBER*						✓			
EDITOR WIKITRUST REPUTATION*						✓	✓		✓
MAX. REVERTS GIVEN AVG. QUALITY*						✓			
IS A BOT*							✓		
EDITOR BLOCKED BEFORE*							✓		
EDITOR'S REVISIONS IN LAST {hour, day, week, month, ever}*							✓		
EDITOR'S EDIT DENSE*							✓		
EDITOR VANDALIZED BEFORE*							✓		
ARTICLE REVISIONS IN LAST {hour, day, week, month, ever}*							✓		
ARTICLE AGE*							✓		
ARTICLE EDIT DENSE*							✓		

Feature	P08	M10	Wa10	We10	A10	A11	W11	J11	Our
COMMENT INDICATES SECTION SPECIFIC							✓		
COMMENT LENGTH W/O SECTION HEADER							✓		
COMMENT INDICATES REVERT							✓		
PREV. USER ANONYMOUS*							✓		✓
TOTAL INSERTED WORDS BY USER*								✓	
TOTAL REMOVED WORDS BY USER*								✓	
TOTAL LOST WORDS FROM USER*								✓	
INSERTING REVISIONS OF USER*								✓	
REMOVING REVISIONS OF USER*								✓	
INSERTING PAGES OF USER*								✓	
REMOVING PAGES OF USER*								✓	
USER HAS A USER PAGE*								✓	
AUTO COMMENT CONTAINS “category”								✓	
AUTO COMMENT CONTAINS “early years”								✓	
AUTO COMMENT CONTAINS “copyedit”								✓	
AUTO COMMENT CONTAINS “personal life”								✓	
AUTO COMMENT CONTAINS “revert”								✓	
REVISION ORDINAL								✓	
REVERTED (MD5 check in window size 10)*							✓	✓	

the 54 features used in their appropriate groups (textual and meta data) with references to their first appearance in the literature and a brief description.

We use the Weka machine learning library (Hall et al. [2009], version 3.7.10) which provides implementations of a number of state-of-the-art machine learning algorithms as classifiers and dataset filters. Furthermore, we employed the Weka extension packages SMOTE, One-class Classifier, and LibSVM.[10]

Figure 2.1 shows the PR curves of the developed binary classifier system using Random Forest. The classifier is evaluated on PAN-WVC-10 and PAN-WVC-11 in comparison to the systems of Mola Velasco [2010] and West and Lee [2011], winners of the 1st and 2nd International Competition on Wikipedia Vandalism Detection, respectively (Potthast and Holfeld [2011]). Our vandalism detection system reaches a PR-AUC of 0.671 on PAN-WVC-10 and 0.748 on PAN-WVC-11.

Meta-data features were shown to be important in improving the recall of classification results. Since our system misses many meta-data features present in Adler et al. [2011], Javanmardi et al. [2011] and West and Lee [2011] it does not reach the performance of the vandalism detection systems created by West and Lee [2011] and Adler et al. [2011], which remain the best performing systems on the PAN-WVC corpora. An additional reason for the poor performance can be badly chosen features to describe a data sample. If a feature does not contribute new information that discriminates vandalism

[10]Weka extension packages can be found on `http://weka.sourceforge.net/packageMetaData`

Table 2.4: *Textual* features used in our vandalism detection system.
The FREQUENCY features are computed for both inserted and removed text.
The IMPACT is computed as ratio of the term frequencies in old and new revision texts. For the used author abbreviations, see Section 2.4.3.

Feature	Prior Use	Description
ALL WORDS FREQUENCY	M10, A11, J11	Frequency of all wordlist words
BAD WORDS FREQUENCY	M10, A11, J11	Frequency of bad words
BIASED WORDS FREQUENCY	M10, A11, J11	Frequency of biased words
MARKUP FREQUENCY	M10, A11, J11	Frequency of Wiki markup words
PRONOUNS WORDS FREQUENCY	P08, M10, A11, W11, J11	Frequency of pronouns
SEX WORDS FREQUENCY	M10, A11, J11	Frequency of sex words
VULGARISM WORDS FREQUENCY	P08, M10, Wa10, W11, J11	Frequency of vulgarism words
ALL WORDS IMPACT	M10, A11, J11	Ratio between frequency of all wordlist words
BAD WORDS IMPACT	M10, A11, J11	Ratio between frequency of bad words
BIASED WORDS IMPACT	M10, A11, J11	Ratio between frequency of biased words
MARKUP WORDS IMPACT	M10, A11, J11	Ratio between frequency of Wiki markup words
PRONOUNS WORDS IMPACT	P08, M10, A11, W11, J11	Ratio between frequency of pronouns
SEX WORDS IMPACT	M10, A11, J11	Ratio between frequency of sex words
VULGARISM WORDS IMPACT	P08, M10, W11, J11	Ratio between frequency of vulgarism words
LONGEST CHAR SEQUENCE	P08, M10, A11, W11, J11	Size of the longest sequence of the same character in inserted text
COMPRESSIBILITY	P08, M10, A11, J11	Ratio of zip-compressed and uncompressed inserted text
AVERAGE TERM FREQUENCY	P08, M10, A11	Average frequency of inserted terms relative to the old revision text
LONGEST WORD	P08, M10, A11, W11, J11	Longest inserted word (links are not considered)
SIZE RATIO	P08, M10, Wa10, A11, J11	Ratio of the old and new revision text size
REPLACEMENT SIMILARITY	P08	Jaro-Winkler Distance of the removed and inserted text
UPPER TO ALL RATIO	P08, M10, A11, W11, J11	Ratio of uppercase to all characters in inserted text
UPPER TO LOWER RATIO	M10, A11, J11	Ratio of uppercase to lowercase characters in inserted text
DIGIT RATIO	M10, A11, J11	Ratio of digits to all characters in inserted text
NON-ALPHANUMERIC RATIO	M10, J11	Ratio of non-alphanumeric to all characters in inserted text
CHARACTER DIVERSITY	M10, J11	Measures how many unique characters are amongst all inserted
SIZE INCREMENT	M10, A11, W11	Number of inserted or removed characters
WORD INCREMENT	–	Number of inserted or removed words
BLANKING	J11	Whether the new revision text size is $<$ 7 characters
INTERNAL LINKS ADDED	J11	Number of internal links in inserted text
EXTERNAL LINKS ADDED	J11	Number of external links in inserted text
ARTICLE SIZE	W11	Size of new revision text
EMOTICONS FREQUENCY	–	Frequency of emoticons
EMOTICONS IMPACT	–	Ratio between frequency of emoticons
UPPERCASE WORDS RATIO	–	Ratio of uppercase words in inserted words
INSERTED CHARACTER DISTRIBUTION	P08, M10, A11, J11	Kullback-Leibler divergence of old revision and inserted text
REMOVED CHARACTER DISTRIBUTION	J11	Kullback-Leibler divergence of old revision and removed text
REVISIONS CHARACTER DISTRIBUTION	J11	Kullback-Leibler divergence of the old and new revision's text

Table 2.5: *Meta data* features used in our vandalism detection system.
Features marked with an asterisk * cannot be computed from the data the PAN training and test corpora are providing and need API calls or additional data of the previous edit such as the `timestamp` or `username`. For the used author abbreviations, see Section 2.4.3.

Feature	**Prior Use**	**Description**
ANONYMITY	P08, M10, A10, A11, W11	Whether the contributor is anonymous
ANONIMITY PREVIOUS	W11	Whether the previous contributor is anonymous
COMMENT LENGTH	P08, M10, We10, A10, A11, W11, J11	Length of the edits comment
TIME OF DAY	We10, A10, A11, W11	The time of day the edit was made on
WEEKDAY	We10, A11, W11	The weekday the edit was made on
EDITS PER USER*	P08, Wa10	The number of edits a user did before on the same article
TIME INTERVAL*	A10, A11, W11	Time from the edit to the previous edit
SAME EDITOR*	A11, W11	Whether the contributor is the same as for the previous edit
USER REPUTATION*	A11, W11	The contributor's average WikiTrust reputation in the new revision text

and regular edits, it might not be well suited to form part of the classification model. On the other hand, a certain combination of features may improve performance more than a single feature alone. The literature on Wikipedia vandalism detection is not often very clear about why certain feature sets are used. However, the problem of the suitability of feature combinations and subsets is well known and often discussed. Javanmardi et al. [2011] tried to find the minimal subset of a number of vandalism features aiming at the highest classification performance. They used the *Least Absolute Shrinkage and Selection Operator* (Lasso, Tibshirani [1994]) on 66 individual features. To shrink the computational effort, they estimated an optimal feature subset of 28 features producing almost the same classification performance as their full feature set on PAN-WVC-10. For our vandalism detection system we considered the resulting subset, but found some of the employed features, such as `AUTO COMMENT CONTAINS "..."` features, not to be performance-improving in our feature set using the PAN-WVC corpora. Furthermore, we found the combined feature `WORDS INCREMENT` to be more decisive than the split-up counterparts `INSERTED WORDS` and `REMOVED WORDS`. Future work to improve our vandalism detection system is the implementation of the remaining meta data features as well as the determination of an optimal feature set.

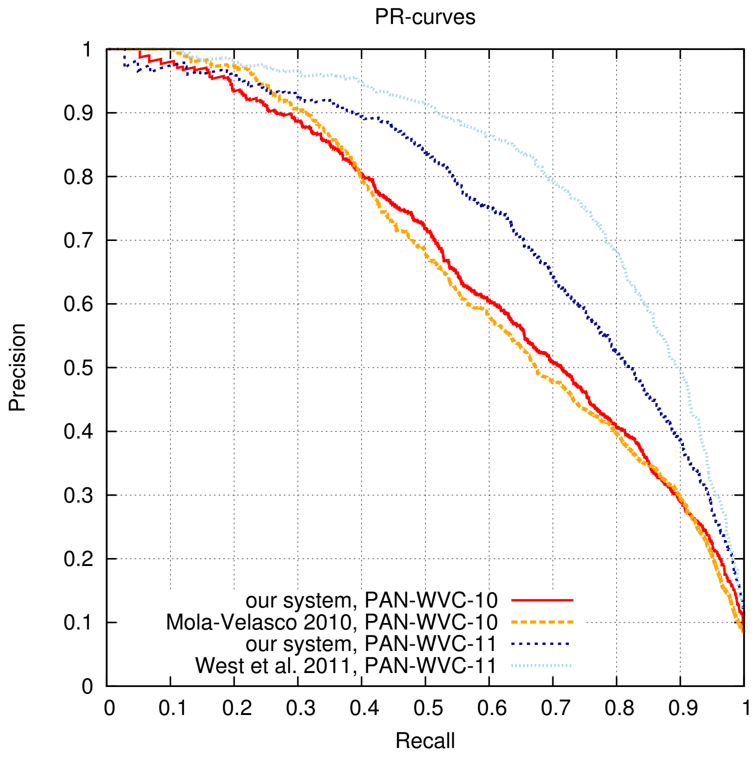


Figure 2.1: PR curves for Random Forest classifier on PAN-WVC-10 and PAN-WVC-11 corpora in comparison to the systems by Mola Velasco [2010] and West and Lee [2011].

Chapter 3

Analyzing The Class Imbalance Problem In Wikipedia Vandalism Detection

Commonly used vandalism corpora, such as the PAN-WVC-10 and the PAN-WVC-11, provide data that is highly skewed; that means it has an imbalanced number of vandalism and regular edits. Only about 6–7% of all samples are annotated as vandalism edits.

Learning traditional classifiers with skewed datasets can lead to poor detection performance. Surveys of classifying imbalanced data by He and Garcia [2009] and Ganganwar [2012] show three reasons for performance decline:

1. If a classifiers learns by minimizing the overall error, then the minority class instances contribute little to the error. This biases the classifier towards the majority class.

2. Many classifiers assume a balanced class distribution of the minority and the majority class, which is not often the case when working with realistic scenarios.

3. Often classifiers implicitly assume equal costs for misclassification for both classes, which is often not sensible: for example the cost for classifying cancer as non-cancer is way higher than the other way round. Since not found cancer data can result in neglected therapy, misclassification can be dangerous to life.

In general, there are two approaches to overcome the class imbalance problem. The first one operates on the data level and comprises several training data resampling techniques. The second approach works on the algorithmic level and involves adjusting the misclassification costs or probabilistic estimates, e.g., at the tree leaves of decision tree classifiers, as well as learning classifier models solely based on minority class samples (*so-called one-class classification*).

In Section 3.1 we examine the impact of training dataset resampling on vandalism detection performance. We find that, in most cases, resampling reduces the performance of the tested classifiers. Logistic Regression, RealAdaBoost and Bayesian Network classifiers benefit from certain resampling strategies, whereas a Random Forest classifier turns out to be relatively unaffected by resampling approaches.

To the best of our knowledge, Wikipedia vandalism detection has not been considered to been identified as a one-class classification problem until now. Therefore, we evaluate the performance of two one-class classifiers in Section 3.2. We observe that the two considered one-class classifiers cannot compete with a Random Forest classifier, which continues to be the best performing binary classifier using the PAN-WVC-10 and PAN-WVC-11 corpora for training and evaluation.

3.1 Evaluating Resampling Approaches

One approach to overcome performance issues of classifiers is resampling the training dataset in order to balance the classes. There are several common approaches to do so, namely random undersampling, random oversampling, directed over- and undersampling and hybrid methods which combine the aforementioned (for a comprehensive overview see He and Garcia [2009]).

3.1.1 Resampling Strategies

Random undersampling (RUS) RUS removes a certain amount of randomly picked majority class instances from the training dataset. RUS leads to class balancing or, , in an extreme case, even to majority class removal. However, a disadvantage of RUS is the loss of possibly decisive instances. Since important information for the class separation is likely to be removed, this technique might induce a lower classification performance.

Random oversampling (ROS) ROS reproduces a certain amount of randomly chosen minority class samples. Thus, the class distribution can be adjusted towards a uniform distribution. Since classifiers, after oversampling, are trained by using some minority class values multiple times, the learned model is likely to overfit. Hence, we prefer the following more sophisticated resampling approach over ROS.

SMOTE The **S**ynthetic **M**inority **O**versampling **TE**chnique (SMOTE) by Chawla et al. [2002] oversamples the minority class by computing artificial instances. The feature values of these samples are calculated by random interpolation of the K-nearest neighbors' feature values (typically $K = 5$). The method aims at avoiding overfitting while oversampling minority class instances.

Since its introduction, SMOTE has experienced multiple improvements. Han et al. [2005] extend SMOTE to use only the minority class samples at the class borderline (*borderlineSMOTE*) in order to generate artificial data which is more important for classification. Maciejewski and Stefanowski [2011] improve this approach by considering the local neighborhood of minority class samples in order to eliminate majority class samples, so that they are not used for data generation. With this approach they attempt to overcome the problem of small disjuncts, where minority class samples are decomposed to multiple small subregions (Jo and Japkowicz [2004]).

Due to the availability as Weka extension package, we use the original SMOTE oversampling approach in our experiments.

3.1.2 The Classifiers

This section briefly describes the classifiers used to evaluate the aforementioned resampling approaches in Section 3.1.3. We focus on Logistic Regression and Random Forest, since they are used by Potthast et al. [2008], Mola Velasco [2010], and Adler et al. [2011]. Additionally, we consider RealAdaBoost as a state-of-the-art Boosting algorithm and a Bayesian Network classifier as a Bayes approach that is reported to outperform the Naive Bayes classifier used by Smets et al. [2008]. Applying Support Vector Machines on the Wikipedia vandalism detection task was pointed out to result in a low classification performance without investing in an appropriate parameter tuning (Mola Velasco [2010]). Thus, we do not take Support Vector Machines into account for our experiments.

Logistic Regression Regression analysis estimates the relationship between a dependent variable and one or more independent variables. In Logistic Regression, the dependent variable represents a class probability. A number of given features form the set of independent variables. To describe their relationship, Logistic Regression uses the logistic function. For each feature, two regression coefficients have to be determined using the maximum likelihood method. The Weka implementation modifies the original Logistic Regression algorithm in order to handle instance weights and improves the classification performance by using a ridge estimator instead of the standard maximum likelihood estimator. This approach, presented by Cessie and van Houwelingen [1992], is reported to reduce overfitting and to perform better in the case of high-dimensional feature sets and highly correlated features.

RealAdaBoost *RealAdaBost* (Friedman et al. [2000]) is a boosting algorithm based on Adaptive Boosting (AdaBoost) by Freund and Schapire [1996]. Boosting is a method to enhance classification performance by combining many weak base classifiers (weak hypotheses) in order to create a more powerful classifier. In AdaBoost, beginning with a start distribution, in which all training samples have the same weights, a first classifier is learned. For learning the next classifier the weight of misclassified samples is increased by a factor depending on the weighted training error. The reweighed samples are used as training data for a refined classifier. This focuses the classifier on hard to detect examples. After a defined number of rounds, the linear combination of the resulting classifiers forms the final classifier.

In contrast to the (discrete) AdaBoost approach, in the RealAdaBoosta algorithm the weak classifiers return a real-valued class probability estimate instead of a discrete class. Each classifier contributes to the final classifier with half the logit-transform of its probability estimate. AdaBoost and RealAdaBoost were empirically found to be relatively resistant to overfitting, whereas, this claim is disputed and precise evidence is lacking (for a comprehensive discussion see Mease and Wyner [2008]).

Bayesian Network A *Bayesian Network* (BayesNet, Pearl and Russell [2001]) is a directed acyclic graph. The nodes in the graph represent random variables, the arcs signify direct correlations between these variables. A Bayesian Network used for classification tasks aims at efficiently modeling the joint probability distribution of a set of random variables in order to predict unknown data samples. Bayesian networks have global semantics, saying that the full joint distribution is the product of all single conditional distributions of each node. Additionally, there are local semantics, expressing that each variable of the network is independent of their non-descendants given the state of its parents. In order to classify a new data sample, the full joint distribution for the sample's features has to determined. We use *Tree Augmented Naive Bayes* (TAN) described by Friedman et al. [1997] for our experiments. In this approach, each attribute in the graph has only the class value and at most one other attribute as parents. Friedman et al. [1997] reported TAN to result in superior classification accuracy regarding Naive Bayes approaches and to be competitive regarding the tree-based classifier C.45.

Random Forest *Random Forest* (Breiman [2001]) is an ensemble learning technique, constructing a defined number of decision tree predictors and combining them to a predictor set (forest). The individual trees are learned from randomly chosen feature subsets and represent independent and identically distributed random vectors. Each tree is grown to full depth (no pruning is applied). To classify a new data sample the final class is determined by the mode of classes that are predicted by the individual trees. The original author reports Random Forest to be insusceptible to overfitting, while increasing the number of trees. However, Segal [2004] and Statnikov et al. [2008] show that Random Forest can overfit for certain class distributions, where trees with a smaller number of splits and/or a smaller number of nodes result in a higher classification performance than unpruned trees. Furthermore, Breiman [2001] describes Random Forest as tolerant against noise and outlier data.

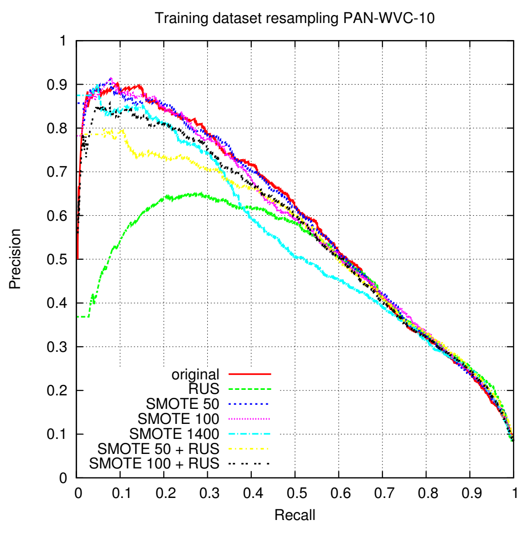
3.1.3 Experiments and Results

We compare the four different classifiers Logistic Regression, RealAdaBoost, BayesNet, and Random Forest regarding their performances using RUS, SMOTE and a combination of both methods.

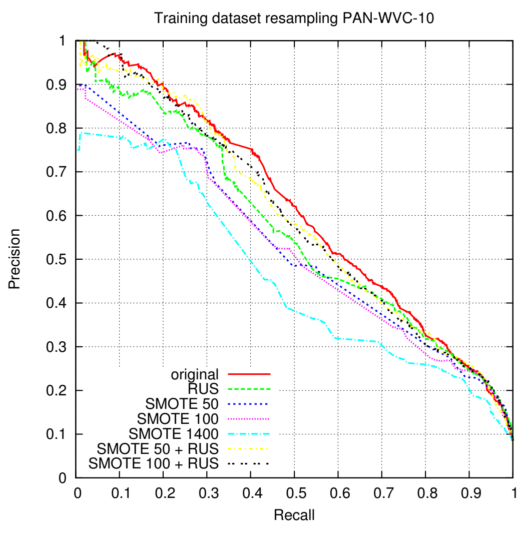
Figure 3.1 shows the PR curves of the test set obtained by an evaluation of the four classifiers on PAN-WVC-10. For SMOTE oversampling we use an amount of 50% and 100% of the original vandalism class instances (denoted by SMOTE 50 and SMOTE 100, respectively). Additionally, we chose an oversampling amount of 1400% which leads to an almost balanced dataset without loss of any majority class instances (13815 (49.53%) vandalism, 14079 (50.47%) regular). Figure 3.2 shows the respective curves for PAN-WVC-11. On PAN-WVC-11 we use a SMOTE oversampling of 1100% instead of 1400%, due to the different class distribution in that corpus (1100% oversampling leads to 28728 (48.88%) vandalism and 30045 (51.12%) regular samples). The classifier parameters used in our experiments can be found in Table A.1.

Table 3.1 provides the corresponding PR-AUC values. Furthermore, Table 3.2 illustrates the performance of the applied resampling techniques in relation to each other. It turns out that, although being relatively insensitive to training data resampling, Random Forest remains the overall best-performing classifier. In what follows, we discuss the evaluated resampling approaches in detail.

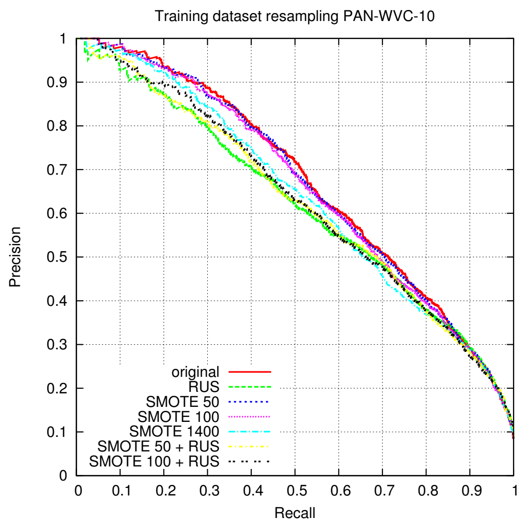
RUS Using random undersampling on the training data, all classifiers but RealAdaBoost on PAN-WVC-11 show a performance drop on both corpora.



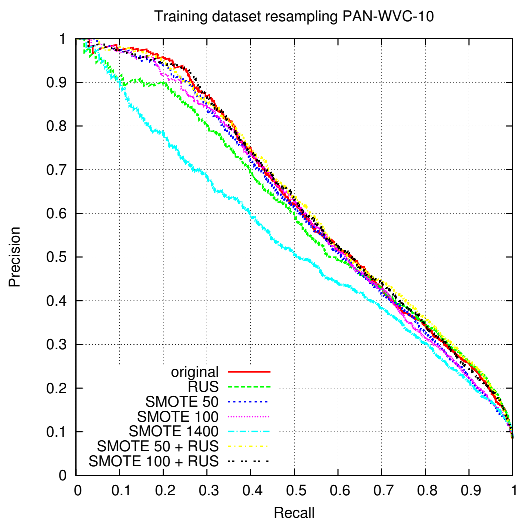
(a) Logistic Regression



(b) RealAdaBoost

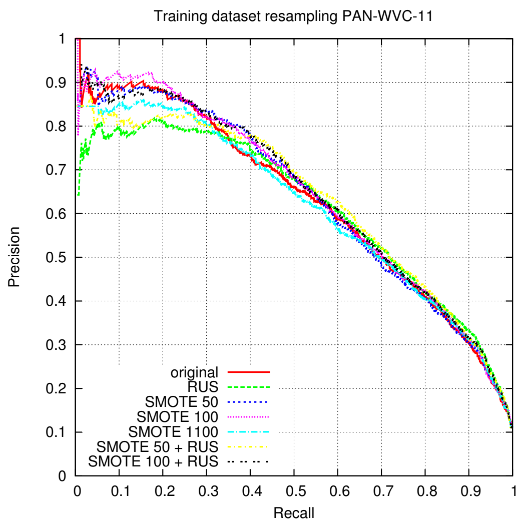


(c) Random Forest

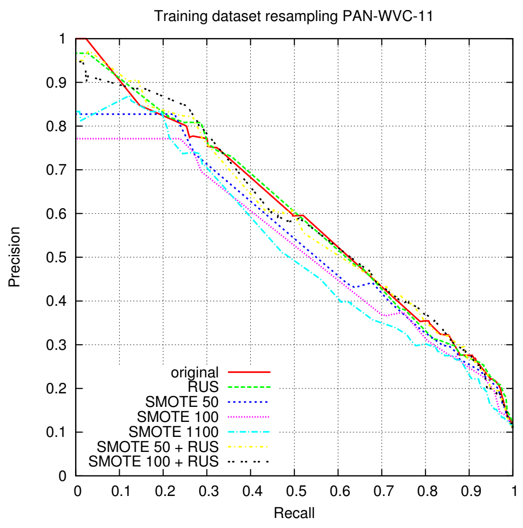


(d) BayesNet

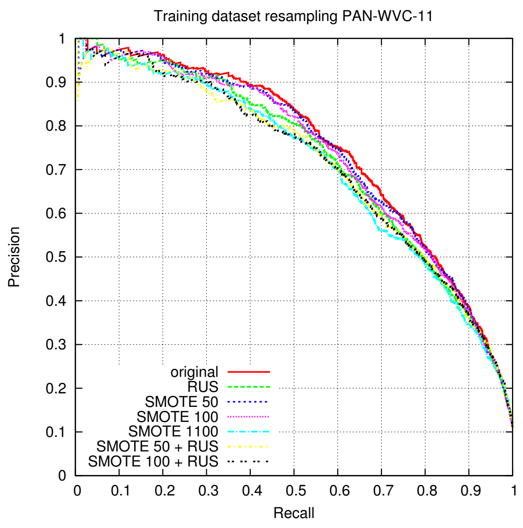
Figure 3.1: Classifier performance (PR-curves) using the resampling approaches RUS, SMOTE and SMOTE + RUS on PAN-WVC-10.



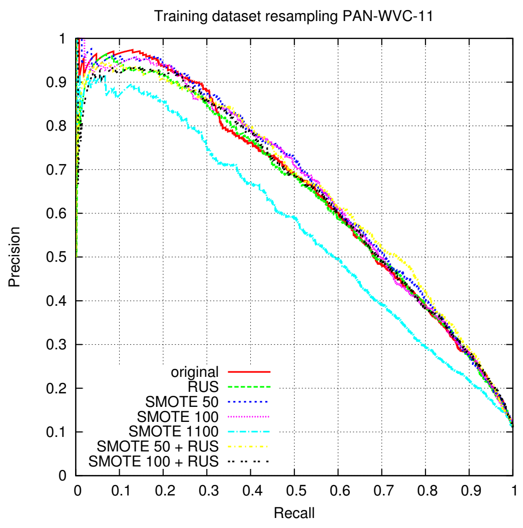
(a) Logistic Regression



(b) RealAdaBoost



(c) Random Forest



(d) BayesNet

Figure 3.2: Classifier performance (PR-curves) using the resampling approaches RUS, SMOTE and SMOTE + RUS on PAN-WVC-11.

Table 3.1: PR-AUC values for the resampling strategies applied to different classifiers. The highest area under curve for each classifier is marked in **bold**.

	Resampling strategies						
Classifier	None	RUS	SMOTE 50	SMOTE 100	SMOTE 1400/1100	SMOTE 50 + RUS	SMOTE 100 + RUS
			PAN-WVC-10				
Logistic Regression	.582	.478	**.584**	.578	.541	.544	.558
RealAdaBoost	**.610**	.561	.475	.486	.443	.598	.598
Random Forest	**.671**	.621	.667	.661	.638	.630	.632
BayesNet	.627	.599	.614	.615	.535	**.633**	.629
			PAN-WVC-11				
Logistic Regression	.631	.614	.636	**.644**	.616	.631	.643
RealAdaBoost	.534	.536	.492	.451	.499	**.564**	**.564**
Random Forest	**.748**	.725	.740	.735	.708	.712	.713
BayesNet	.652	.642	**.664**	.657	.566	.658	.646

Table 3.2: Overview of the relative classification performance using different resampling strategies. From left to right, the strategies lead to a higher performance. For comparison, the best results of each applied strategy are used.

Classifier	**Resampling strategies**						
				PAN-WVC-10			
Logistic Regression	RUS	<	SMOTE + RUS	<	None	<	SMOTE
RealAdaBoost	SMOTE	<	RUS	<	SMOTE +RUS	<	None
Random Forest	RUS	<	SMOTE + RUS	<	SMOTE	<	None
BayesNet	RUS	<	SMOTE	<	None	<	SMOTE + RUS
				PAN-WVC-11			
Logistic Regression	RUS	<	None	<	SMOTE + RUS	<	SMOTE
RealAdaBoost	SMOTE	<	None	<	RUS	<	SMOTE + RUS
Random Forest	SMOTE + RUS	<	RUS	<	SMOTE	<	None
BayesNet	RUS	<	None	<	SMOTE + RUS	<	SMOTE

For Logistic Regression (on both corpora) and Random Forest (on PAN-WVC-10) RUS even leads to the lowest overall performance.

If a classifier already handles class imbalance internally, RUS only removes majority class data that is needed to train the model without benefiting from a balanced data set. Thus, important training data gets lost and less decisive samples are used to train the model which entails under-fitting. For RealAdaBoost the loss of regular samples seems not to be as influential as the training on a balanced dataset.

SMOTE Logistic Regression benefits from SMOTE 50 (on PAN-WVC-10) and from SMOTE 100 (on PAN-WVC-11). Both oversampling strategies result in the best overall performances on the respective corpora. SMOTE oversampling leads to a high performance drop for the RealAdaBoost classifier. For the Random Forest classifier on both corpora, oversampling the target class with SMOTE causes a slight decrease of performance. The performance for BayesNet increases for lower oversampling proportions (50% and 100%) on PAN-WVC-11, SMOTE 50 even leads to the highest overall performance. On PAN-WVC-10 all SMOTE proportions result in a performance drop for the BayesNet classifier.

On PAN-WVC-10 for Logistic Regression and Random Forest, a higher oversampling proportion leads to lower performance. This is also the case for all classifiers but Logistic Regression on PAN-WVC-11. For all classifiers on both corpora SMOTE 1100/1400 lead to the lowest classification performance using SMOTE approaches. An exception is the RealAdaBoost classifier (on PAN-WVC-11), for which SMOTE 1100 outperforms the other proportions.

A reason for the observed lower performance using SMOTE might be the absence of significant data in the training and test corpora. If the vandalism samples given in the test dataset represent other vandalism types than those given in the training set, some kinds of vandalism will never be found. Wikipedia vandalism has been found to be a heterogeneous problem (Chin and Street [2011] identified 11 different vandalism categories). Hence, an underrepresentation of vandalism edits from certain categories in the training corpora would not be surprising, since the samples have been chosen randomly (Potthast [2010], Potthast and Holfeld [2011]). In the case of missing decisive vandalism samples, oversampling would not produce a more accurately defined vandalism class region, but would only insert further weak samples. This is especially the case if the classes of vandalism edits and regular edits overlap in their feature spaces. The classes' borders are then further softened by inserting in-between data by oversampling.

SMOTE + RUS SMOTE + RUS outperforms the other resampling strategies for RealAdaBost (on both PAN-WVC corpora) and for BayesNet (on PAN-WVC-10). Using the SMOTE + RUS approach even leads to the highest overall performance for RealAdaBoost on PAN-WVC-11 and for BayesNet on PAN-WVC-10 (SMOTE 50 + RUS). Using RealAdaBoost, on both corpora the classification performance is independent of the applied oversampling proportion. Applying a higher oversampling proportion leads to a higher classification performance for all classifiers but BayesNet and RealAdaBoost. Using

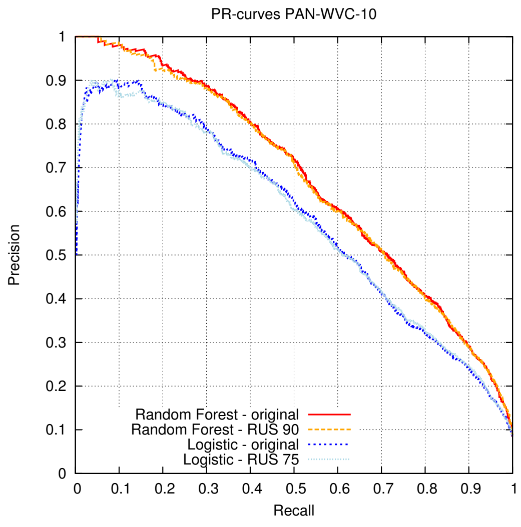
Random Forest, SMOTE + RUS results in the lowest overall classification performance on PAN-WVC-11.

Although SMOTE seems to insert weak samples into partly noisy areas, the combination with RUS generally leads to a higher performance than only applying RUS, because of the additional majority class samples. On the other hand, the superiority of the SMOTE-only approach for all classifiers but RealAdaBoost on both corpora and BayesNet on PAN-WVC-10, leads us to the assumption that the absence of majority training data is more damaging than the insertion of possibly weak data into noisy regions.

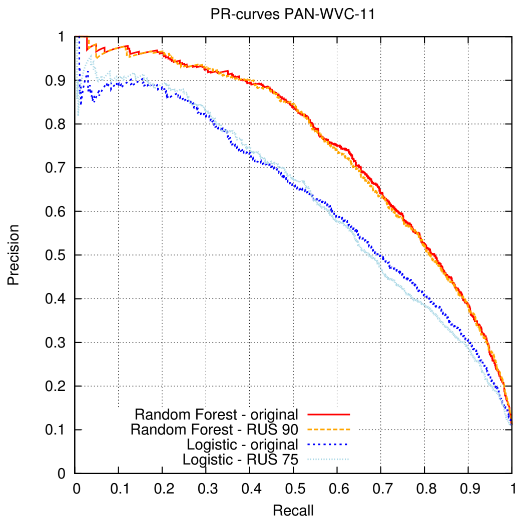
A reason for not increasing the overall performance can be the not optimally chosen oversampling and undersampling proportions. Estabrooks and Japkowicz [2001] show that optimal oversampling and undersampling ratios are problem-dependent and have to be estimated for each data set. Thus, the combination of oversampling with a certain undersampling proportion might be essential to observe the often mentioned leverage of SMOTE resampling on the classification performance and needs further investigation for the PAN-WVC corpora. Especially for overlapping class regions, RUS might remove majority samples from noisy areas. Thus, an improved oversampling approach might be able to insert additional valuable data in these regions.

Van Hulse et al. [2007] provide a comprehensive overview of using resampling techniques in combination with different classifiers learned on unbalanced datasets. They report random undersampling to be the best-performing resampling technique. For Random Forest with a class distribution of 5-10 % minority class samples, Van Hulse et al. observe RUS 90 to be the best-performing technique, better than without resampling. For Logistic Regression RUS 75 is reported to perform better than the original classifier. Both results could not be confirmed on the PAN-WVC corpora (see Figure 3.3 and Table 3.3 for corresponding PR-AUC values). In contrast, we found SMOTE to lead to the best performances for Logistic Regression. On both corpora, Logistic Regression using RUS 75 results in nearly the same PR-AUC, however, slightly higher precisions appear for lower recall and vice versa. Random Forest appears to be nearly insensitive to the given RUS proportion. Applying Random Forest using RUS 90 results in a minimal lower performance on both PAN-WVC corpora.

Summarizing our experiments, we can conclude that RealAdaBoost is most affected by training data imbalance. Ovserved results, such as the best overall performance for SMOTE + RUS (on PAN-WVC-11) and the superior classification performance using (nearly balanced) SMOTE 1100 over SMOTE 50 and SMOTE 100, underpin this statement. Random Forest shows only little sensitivity to resampling approaches. However, it turns out to be the best



(a) PAN-WVC-10



(b) PAN-WVC-11

Figure 3.3: Classifier performance (PR-curves) on the PAN-WVC copora using optimal resampling approaches reported by Van Hulse et al. [2007].

performing classifier of all evaluated approaches without applying resampling strategies.

A number of investigations found that the class imbalance problem alone cannot be considered to be the only reason for a low classification performance. Japkowicz [2001] and Jo and Japkowicz [2004] identify the imbalance problem to lead to small inter-class disjuncts (sub-clusters) which cause the performance drop. As a means to improve the performance, they propose considering both *between-class* imbalance (majority-minority class count difference) and *within-class* imbalance (different number of the samples within a class's sub-clusters). Moreover, Denil and Trappenberg [2010] show that the impact

Table 3.3: PR-AUCs for optimal resampling strategies as reported by Van Hulse et al. [2007]. The highest PR-AUC for each classifier on the respective corpus is marked in **bold**.

Classifier	**Logistic Regression**		**Random Forest**	
Corpus	original	RUS 75	original	RUS 90
PAN-WVC-10	**.582**	.580	**.671**	.668
PAN-WVC-11	**.631**	.630	**.748**	.745

of the imbalance problem declines, when larger datasets are used. Similar to García et al. [2007], they show the class overlap to be more corruptive than the class imbalance problem. To tackle the described objects, Batista et al. [2004] propose to use SMOTE in combination with data cleaning techniques such as Tomek links (Tomek [1976]) and Edited Nearest Neighbor Rules (ENN) (Wilson [1972]). Another approach to clean up imbalanced datasets with a neighborhood cleaning rule (NCL) based on ENN was proposed by Laurikkala [2001]. More investigation is needed to point out the within-class imbalance properties in the PAN-WVC corpora regarding certain feature sets. Considering the mentioned advanced dataset cleaning methods might be a first step to improve the classification performance on the PAN-WVC corpora.

3.2 One-class Classification

A second approach to overcome poor classification performance due to class imbalance is *one-class classification*. This section introduces the one-class classification problem. For the first time, we apply two one-class classifiers to the problem of vandalism detection and evaluate them on the PAN-WVC corpora. Attempt are made to provide reasons for the obtained poor classification performances. Both classifiers were found not to be competitive with the considered two-class classifiers using the PAN-WVC training and test data sets.

3.2.1 The One-class Classification Problem

One-class classification methods learn a classification model by only considering values of a single class of interest – the so-called target class. Such an approach may be advantageous if a second class's data is not available or rather hard to collect. Additionally, if the training data set is large and highly skewed, one-class classification approaches might drastically speed up the learning procedure by considering less training samples. Other terms for one-class classification are outlier detection, novelty detection or concept learning (Tax [2001]).

The aim of one-class classification is to create a distinct region around the target class, so that as many target class samples as possible and only a minimal number of outlier class samples are located inside of this region. Only considering a target and one outlier class can be advantageous in the case of multi-class problems, where it is only important, whether a sample belongs to the target class, or not. All non-target class samples are then combined into one outlier class.

3.2.2 Approaches to One-class Classification

One-class classification approaches can be divided into three main methods, namely *density methods*, *reconstruction methods* and *boundary methods* (Khan and Madden [2010], Mazhelis [2006], Tax [2001]).

Density methods Densitiy methods are based on an estimation of a probability density function (PDF) of all features of the target class samples. If information about the outlier class is missing, its PDF is assumed to be uniform. In order to classify a new data sample, the values of its feature vector are compared to a simple threshold. Examples for density methods are Gaussian and Mixture of Gaussian, Markov models, Parzen density estimation, and K-nearest-neighbors. The one-class classifier of Hempstalk et al. [2008] uses a density method in the first step of their approach (see the following section).

Reconstruction methods Reconstruction methods make assumptions on the underlying data structure. A basic model is build based on multiple training data samples, so that the model provides an optimal representation of the feature vectors in the training samples. In order to classify new data samples, the model's reconstruction error for a feature vector is minimized. Examples of reconstruction methods are K-means, self-organizing maps, Principle Component Analysis, autoencoders, association rule-based classification, and time series analysis.

Boundary methods Boundary methods build a boundary around the training data vectors. In order to classify new data , the distance of the new data sample's feature vector to the boundary is considered. Examples for boundary methods are K-centers, Support Vector Data Description (SVDD) and ν-Support Vector classification. The used one-class SVM approach by Schölkopf et al. [1999] can be assigned to the ν-Support Vector classification methods.

3.2.3 The One-class Classifiers

In this section we briefly describe the one-class classifiers that are employed in our evaluations in Section 3.2.4.

One-class classifier by Hempstalk et al. [2008] Hempstalk et al. [2008] propose a one-class classifier that applies a supervised approach (two-class classification) to an unsupervised learning problem (one-class problem). More precisely, their approach combines a density estimator with a class probability estimator (two-class classification algorithm).[1] First, they use a one-class density method and apply a density estimator to a one-class training dataset. The obtained probability distribution for the target class is used as reference distribution to generate artificial data using a Gaussian data generator. The artificial data forms a second class (outlier). The target and outlier classes build the final training dataset for an arbitrary two-class classifier, which can then be used to classify new data samples.

One-class classification with LibSVM A Support Vector Machine (SVM), originally presented by Boser et al. [1992], maps training data into a feature space and separates classes via a hyperplane. The hyperplane is described by a subset of the feature vectors (support vectors) and is built by maximizing the distance of the nearest feature vectors to the hyperplane. Thus, SVMs are also called maximum margin classifiers. To enable building a model on non-linearly separable datasets, the so-called *kernel-trick* is used. By using non-linear kernel functions, the data is mapped into a feature space with more dimensions in order to allow for finding a hyperplane that separates the classes. After retransforming the found hyperplane to the lower dimensional space, the hyperplane appears to be non-linear or even non-contiguous.

Schölkopf et al. [1999] extended SVMs to create a one-class SVM approach which only considers the samples of a target class to train a classification model. In the one-class SVM approach, a second class for training is missing. Thus, instead of building a hyperplane to separate two classes with maximum margin, the hyperplane is built by determining the maximum margin for separating the samples from the *origin* of the feature space.

3.2.4 Experiments and Results

We apply both of the aforementioned one-class classifiers on the PAN-WVC corpora. In this section we report on the obtained results and attempt to explain the poor performances that both approaches achieve when evaluating them on the PAN-WVC test sets.

One-class Classifier (Hempstalk et al. [2008]) There are a number of parameters that can be adjusted in the Weka implementation of the one-class classifier, such as the proportion of generated artificial data relative to the number of vandalism samples, the number of iterations and left-out data proportion, used for internal cross validation, and, whether to use instance weights

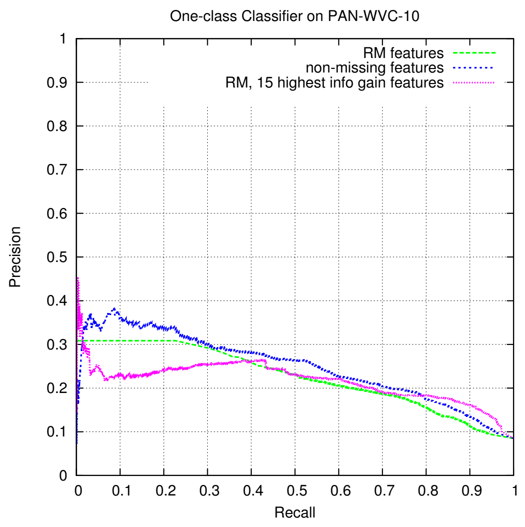
[1]An implementation of the one-class classifier as Weka extension package is available on `http://weka.sourceforge.net/packageMetaData/oneClassClassifier/`

and Laplace correction. Experimenting with different parameter setups, e.g. by increasing the iteration number and proportion of generated artificial data, we were not able to find a setup that resulted in an improved performance. Thus, we used the default values for the parameters and the default Bagging (Breiman [1996]) with REPTrees classifier as the two-class classifier. We increased the number of iterations that are used for Bagging to 100. Since Random Forest performs best in our experiments in Section 3.1.3, we also considered Random Forest as two-class classifier, but found it to result in a lower classification performance regarding the Bagging approach. The parameter settings for this classifier can be found in Table A.1.

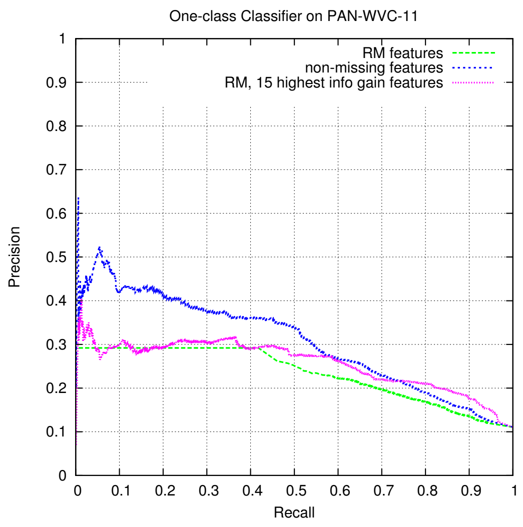
While computing the features for PAN-WVC-10 and PAN-WVC-11, some values are missing in the training and test data features. For features such as `EDITS PER USER` some pages are not available, anymore, in the Wikipedia history. Some features use the clean text, which is extracted from the original Wikipedia markup text. In case the markup was not extractable by our Wikitext extractor, a missing value was inserted instead. Features such as `INSERTED CHARACTER DISTRIBUTION` and `REMOVED CHARACTER DISTRIBUTION` return missing values if no text is inserted and removed, respectively. While the binary classifiers evaluated in Section 3.1.3 handle missing data by replacing it with mean values (see Weka's `weka.filters.unsupervised.attribute.ReplaceMissingValues` filter), the one-class classifier does not handle missing data internally. We obtained a very poor classification performance, when using the original training data including missing values. Thus, in a preprocessing step, we replace missing values in the training dataset by using Weka's `ReplaceMissingValues` filter.

We consider three different feature sets to form the training data. First, we use the full feature set described in Section 2.5 replacing missing attribute values. Second, our feature set comprised only features without missing attribute values. Since there are some features having only one missing value, we replace them by mean values, and, however, take these features into account. As a third feature set, we determine the 15 highest ranked features using Weka's information gain attribute selector. The specific features that are used in the non-missing and highest info gain training datasets are given in Section A.2. Figure 3.4 shows the PR curves obtained by evaluating the one-class classifier on the PAN-WVC corpora. The respective PR-AUC values are given in Table 3.4.

Using training data comprising only features without missing values leads to the highest classification performance on both PAN-WVC corpora. With respect to using all features, training the classifier with the 15 highest info gain features increases the precision for recall values higher than 0.4, and keeps or increases the precision for lower recall values on PAN-WVC-11 and PAN-WVC-



(a) PAN-WVC-10



(b) PAN-WVC-11

Figure 3.4: PR curves for the one-class classifier by Hempstalk et al. [2008] on PAN-WVC-10 and PAN-WVC-11. RM denotes **R**eplaced **M**issing feature values.

Table 3.4: PR-AUC values for different feature sets using the one-class classifier by Hempstalk et al. [2008]. The highest PR-AUC on each corpus is marked in **bold**.

	Feature set		
Corpus	replaced missing	non-missing	15 highest info gain
PAN-WVC-10	.224	**.249**	.216
PAN-WVC-11	.233	**.306**	.256

10, respectively. On PAN-WVC-11, both approaches using reduced feature sets result in higher classification performances than using the full feature set. This shows that for achieving an optimal classification performance, the chosen feature set plays an important role. The reasons for the poor performance on certain feature sets may be found in the particular components of the one-class classifier. It seems as if the density estimator, that is applied to the target class training samples, is not able to produce a proper reference distribution. Hence, the created artificial class samples do not cover the actual feature value distribution of the regular class samples from the test dataset. An improper reference distribution can result from a too complex feature distribution to learn on or from overlapping feature spaces of the target class and the outlier class.

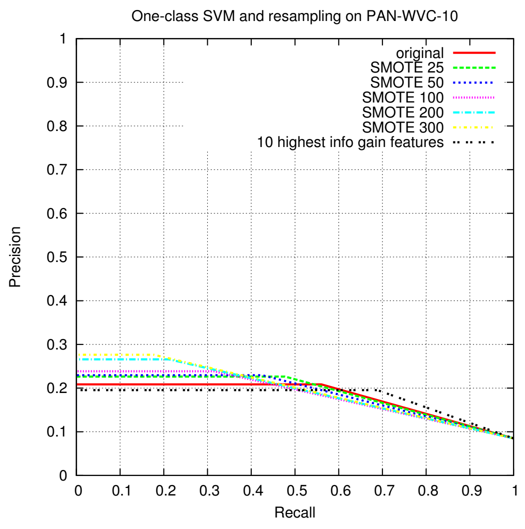
Since the precision never appears to be higher than 0.4 for a recall of at least 0.3, the trained one-class classifier cannot be considered to be a proper approach to solve the vandalism detection task, given the considered options and feature sets. Improvements can be made by determining the optimal set of highly discriminative features. Furthermore, inserting mean values for missing attribute values may result in inappropriate values in a data sample, which do not reflect real vandalism edits and regular edits properly. Since we only evaluate Bagging and Random Forest, the impact of different two-class classifiers on the one-class classification performance should be investigated in future work.

One-class SVM Since it is recommended to apply SVMs on normalize data (Graf and Borer [2001]), we normalized the training and test data attribute values for both PAN-WVC corpora in a preprocessing step. Again, a number of parameters can be modified for the one-class SVM, such as the used kernel function and its function parameters, whether to replace missing values and the parameter ν, which characterizes the fraction of support vectors and the expected proportion of outliers. By experimenting, we found a value of $\nu = 0.29$ to increase the performance on both PAN-WVC corpora regarding the default value of 0.1. For the remaining parameters we used the provided default values. The classifier options that were employed in our experiments are shown in Table A.1.

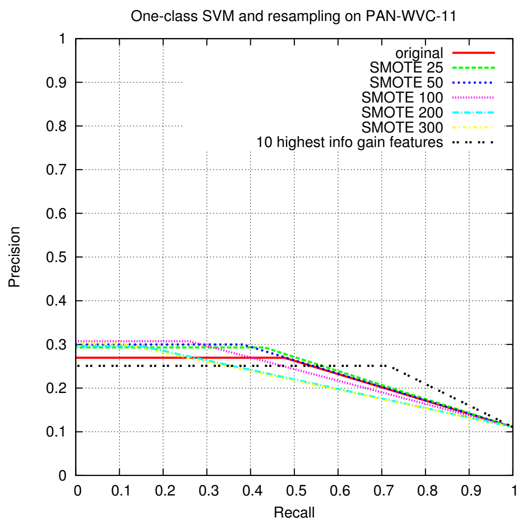
The one-class SVM classifier returns the determined class label instead of a class probability value. This fact exacerbates a direct comparison of the classifiers' performance with PR-AUC values of different classifiers. It also implies that the classifier's classification properties cannot be adjusted at classification time, but have to be determined by its parameters during training.

We experiment with five different SMOTE proportions (25, 50, 100, 200, and 300% oversampling). SVMs are reported to suffer from the curse of dimensionality (Evangelista et al. [2006]). This means that a high dimensional feature space can lead to worse classification performances than building the model on a lower number of features. Hence, we additionally select only the 10 highest ranked features using Weka's information gain attribute selector. The resulting feature set is given in Section A.2.1. Figure 3.5 shows the PR curves for applying the LibSVM one-class classifier on the PAN-WVC corpora. The respective PR-AUC values are given in Table 3.5.

The overall classification performance of the one-class SVM classifier on both PAN-WVC corpora turns out to be rather low. Oversampling the training data with SMOTE causes a drop of recall and increase of precision. The



(a) PAN-WVC-10



(b) PAN-WVC-11

Figure 3.5: PR curves for one-class SVM on PAN-WVC-10 and PAN-WVC-11.

Table 3.5: PR-AUC values for different feature sets using one-class SVM as returned by the Weka implementation. The highest PR-AUC on each corpus is marked in **bold**.

	Resampling/Feature set						
Corpus	original	SMOTE 25	SMOTE 50	SMOTE 100	SMOTE 200	SMOTE 300	10 highest info gain
PAN-WVC-10	.154	.152	.146	.133	.123	.119	**.161**
PAN-WVC-11	.186	.189	.183	.162	.138	.130	**.211**

higher the oversampling proportion on PAN-WVC-10, the higher the precision and the lower the recall. On PAN-WVC-11, the precision almost remains constant, while the recall drops for higher oversampling proportions. Learning the one-class SVM with the reduced feature set results in the highest classification performance on both PAN-WVC corpora. This observation supports the aforementioned sensitivity of SVM approaches regarding the number of features. The smaller feature set leads to a slight decrease of precision, but on the other hand, to a significant increase of recall.

This means that determining an optimal feature set is a promising option to increase the classification performance of the one-class SVM. Furthermore, as proposed by Hempstalk et al. [2008], the one-class SVM classifier's comparability could be improved by tuning the parameters of the classifier. To compare their density-based one-class classifier with the one-class SVM, they determine the parameter $\nu = 0.1$ and adjusted the kernel functions' parameter γ, so that the false positive rate was as close as possible to 0.1.

A problem for both the one-class classifiers by Hempstalk et al. [2008] and the one-class SVM is their poor precision. While for each classifier a setup can be determined, so that most vandalism cases are found, the large number of false positives (regular edits that are detected as vandalism) let us judge the classifiers not to be suitable for the vandalism detection task, given our feature sets and the corpora we operate on. The inability to discriminate regular edits and vandalism edits after learning only from vandalism samples leads to the assumption, that either the vandalism training data does not describe the real vandalism distribution or that the feature values of the regular edits heavily overlap with the values which represent vandalism. Similar to our suggestions in Section 3.1.3, advanced data clean up methods, together with an optimized feature set, might increase the overall classification performance.

3.3 Conclusions

We compare different resampling strategies applied on four classifiers, namely Logistic Regression, RealAdaBoost, BayesNet and Random Forest. We observe the examined resampling strategies (RUS, SMOTE, and SMOTE + RUS) at the tested proportions to partly increase the classification performance for all tested classifiers but Random Forest. However, regarding the total classification performance, Random Forest, trained with the original data set, outperforms all other approaches. We argue that reasons for the poor improvement by resampling techniques can be found in the class overlapping and/or in-between class imbalance of the PAN-WVC training datasets, given our chosen feature set. Further investigations on the impact of advanced data cleaning and resampling techniques, as well as determining more discriminative features will help to shed light on these issues.

We considere Wikipedia vandalism detection as one-class classification problem and evaluate the one-class classifier by Hempstalk et al. [2008] and the one-class SVM by Schölkopf et al. [1999], as implemented by Chang and Lin [2011], on the PAN-WVC corpora. Given our setup, both one-class classifiers cannot compete with Random Forest or the other binary classifiers. Again, we see the problem of low performance in a poor feature set resulting in low decisive data descriptions for vandalism cases. Additionally, we believe the problem is aggravated by the heterogeneous representation of vandalism on Wikipedia and overlap of vandalism edits and regular edits in the feature space.

Chapter 4

Developing Content Based Classifiers

Wikipedia vandalism corpora, such as PAN-WVC-10 and PAN-WVC-11, provide fully human-annotated datasets. In order to learn a vandalism classification model, it seems to be beneficial to use more dynamic training data to keep detection accuracy up with possibly changing vandalism behavior.

This chapter provides a pilot study on compiling vandalism classifiers learned on so-called article-relative datasets. An article-relative dataset is built on vandalism edits extracted from a Wikipedia article's full edit history. Furthermore, in order to proof the practicality of a category-relative classifier approach, in Section 4.1, we investigate the similarity of vandalism in article-relative datasets. Therefore, we evaluate 12 datasets which we assign to four different article categories. In Section 4.2, we combine multiple classifiers learned on article-relative data and examine the impact of combining multiple article-relative datasets as well as the training dataset of PAN-WVC-11 and article-relative datasets to learn a classifier. Note that in the following sections we denote a classifier that is trained on an article-relative dataset as *article-relative classifier*.

To run our experiments, we developed a Wikipedia vandalism analysis system.[1] The system depends on our Wikipedia vandalism detection system (see Section 2.5) and is based on Apache Hadoop.[2] Our JRuby implementation allows rapid experimental setups by using Hadoop jobs on Wikipedia history dump files.

[1]Our analysis system is available at
`https://github.com/webis-de/wikipedia-vandalism-analyzer`

[2]`http://hadoop.apache.org`

4.1 Simple Article-based Classifiers

Considering the full revision history of an article is not a new concept in the Wikipedia vandalism detection literature. Chin et al. [2010] are the first to use the complete revision history of two Wikipedia articles to create a vandalism detection system based on an active learning approach using language model statistics. West et al. [2010] use revision histories of articles to compile a large experimental datasets from rollback edits (administrative reverts). Chin and Street [2012] elaborate the problem of lacking labeled training data for the many heterogeneous types of vandalism. Using the revision history of Wikipedia articles, they examine how to transfer a classifier learned on a Wikipedia article in order to detect vandalism in another article. Additionally, lots of meta data features, such as reputation features, which are used by West et al. [2010], Adler et al. [2010] and others are based on an analysis of revision histories of Wikipedia articles.

Simple vandalism From the top 1000 English Wikipedia pages with most revisions[3] we select 12 pages that can be found in the `history1` dump files. Similarly to West et al. [2010], who examine an article's revision history to identify vandalism edits, we used the articles' full revision history to automatically compile a training dataset for each article. Instead of analyzing the revision comments regarding rollbacks, we extract the reverted revisions out of each selected article by comparing the SHA1 hash values of revision triples. A revision can be seen as reverted, if the previous and following revision have the same SHA1 hash values. In case of edit wars, in which two or more users revert their edits multiple times, we drop reverted revisions which have the same SHA1 hash as the one before the previous revision. This additional condition avoids detecting regular edits as reverted for some cases of edit wars. The resulting reverted revisions are naively assumed to be vandalism, so that we henceforth call them "simple vandalism".

Two problems appear with this approach. First, overwritten, but not reverted vandalism edits cannot be found, since they provide another SHA1 hash than a full revert to a previous version. This is the case if an intermediate edit adds regular text to vandalized content or if several consecutive revisions are vandalized. One way to tackle this problem can be to consider a larger number of revisions to look for identical text hash values. Secondly, reverts of regular edits are tagged as revisions which are reverted for vandalism purposes, since only the first reverts are considered to be simple vandalism. Even though the

[3]`https://en.wikipedia.org/wiki/Wikipedia:Database_reports/Pages_with_the_most_revisions`, from 03. November 2013, called in Aug. 2014

Table 4.1: Extracted Wikipedia pages and their proportion of simple vandalism. Note that edit war revisions are discarded, thus, each page content (SHA1 hash) appears only once in the compiled corpora. The percentage of simple vandalism is given for each article (left), as well as each category (right).

Title	Id	Total Revisions	Regular Edits	Vandalism Edits	Simple Vandalism %		Category
Anarchism	12	17000	14864	1459	9.8	17.1	Human Concepts (HP)
Capitalism	5416	13088	10691	1982	18.5		
Democracy	7959	13118	10231	2666	26.0		
Batman	4335	12040	9936	1913	19.3	23.8	Fictional Characters (FC)
Doctor Who	8209	17360	13300	3620	27.2		
Abraham Lincoln	307	14904	12388	2228	18.0	16.5	Popular People (PP)
Arnold Schwarzenegger	1806	9630	8028	1456	18.1		
Aaliyah	2144	9508	8329	1071	12.9		
Berlin	3354	10057	8486	1330	15.7	13.9	Geographical Places (GP)
Detroit	8687	11913	10142	1610	15.9		
Afghanistan	737	11585	10120	1107	10.9		
Brazil	3383	14481	12619	1702	13.5		

next revision reverts the revert of regular content, it will be considered to be vandalism.

Table 4.1 overviews the selected articles with their respective proportions of simple vandalism. We assign the used articles to four rough categories: *Human Concepts*, *Fictional Characters*, *Popular People* and *Geographical Places*. Note that these categories do not necessarily correspond to the categories that are assigned to the articles at Wikipedia.

We observe a high percentage of simple vandalism in all articles. Since the percentages of vandalism significantly differ from the values of 2–7% reported in the literature (see Chapter 1), we can assume a higher rate of false positives among the simple vandalism cases extracted from each article's revision history. On the other hand, the most often edited articles, to which the examined ones belong, might be more often vandalized. Thus, an examination of the all Wikipedia articles regarding the number of reverts relative to the article size would be of interest for future work.

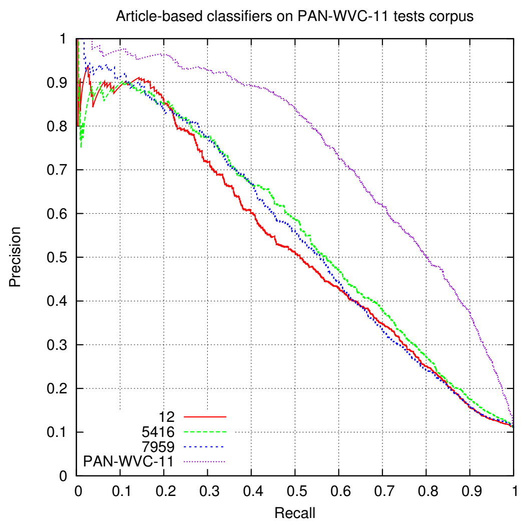
To examine the accuracy of the internal data of our simple vandalism datasets, we apply stratified ten-fold cross validation. With this method, the data set is split into k parts (in our case $k = 10$). For each of k folds, $k - 1$ parts form a training dataset which is used to learn a classifier. The classifier is then evaluated using the kth part as test dataset. Note that each fold uses another part as test dataset. For stratified cross validation, the class distri-

Table 4.2: PR-AUC values of article-relative classifiers. The articles in the *first column* are used as training dataset. The *first row* denotes the respective test data set. The *diagonal*, gray shaded values represent the results of 10-fold cross validation for the certain classifier. The highest classification performances using the PAN-WVC-11 corpora are marked in **bold**.

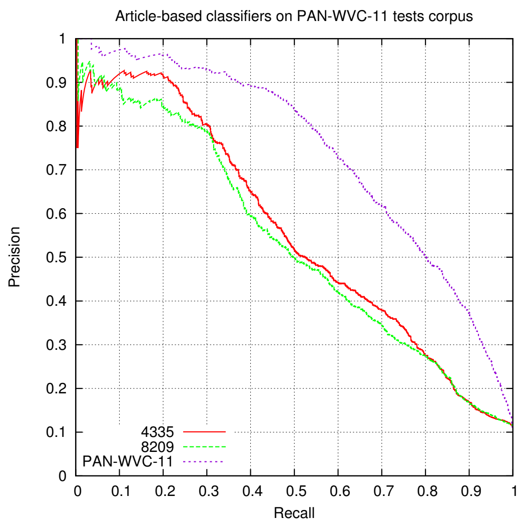
Id	**12**	**5416**	**7959**	**4335**	**8209**	**307**	**1806**	**2144**	**3354**	**8687**	**737**	**3383**	**PAN11**	**Title**
12	.494	.671	.755	.640	.684	.691	.647	.466	.627	.635	.547	.600	.528	Anarchism
5416	.463	.702	.770	.648	.696	.710	.656	.481	.641	.648	.554	.625	.555	Capitalism
7959	.454	.686	.783	.637	.694	.716	.654	.437	.645	.635	.560	.623	.547	Democracy
4335	.455	.661	.752	.700	.697	.716	.665	.482	.615	.643	.529	.614	.558	Batman
8209	.442	.671	.754	.646	.735	.705	.653	.467	.647	.646	.540	.605	.533	Doctor Who
307	.420	.661	.759	.636	.673	.742	.662	.458	.605	.615	.574	.605	.512	Abraham Lincoln
1806	.427	.663	.749	.643	.691	.714	.684	.452	.616	.627	.544	.609	.578	A. Schwarzenegger
2144	.427	.651	.734	.640	.681	.699	.641	.520	.620	.622	.529	.600	.543	Aaliyah
3354	.434	.659	.754	.624	.694	.695	.650	.448	.671	.636	.545	.609	.561	Berlin
8687	.436	.664	.749	.635	.693	.696	.645	.452	.631	.664	.548	.616	**.594**	Detroit
737	.438	.662	.752	.641	.683	.712	.660	.459	.632	.635	.552	.611	.572	Afghanistan
3383	.440	.675	.765	.645	.696	.713	.669	.458	.642	.642	.563	.634	.559	Brazil
PAN11	.364	.587	**.655**	.566	.632	.615	.592	.380	.548	.581	.468	.521	.739	
	Anarchism	Capitalism	Democracy	Batman	Doctor Who	Abraham Lincoln	A. Schwarzenegger	Aaliyah	Berlin	Detroit	Afghanistan	Brazil		

bution of each partial dataset is the same as the distribution in the original dataset. Finally, the total performance is obtained by averaging the individual performances of each fold.

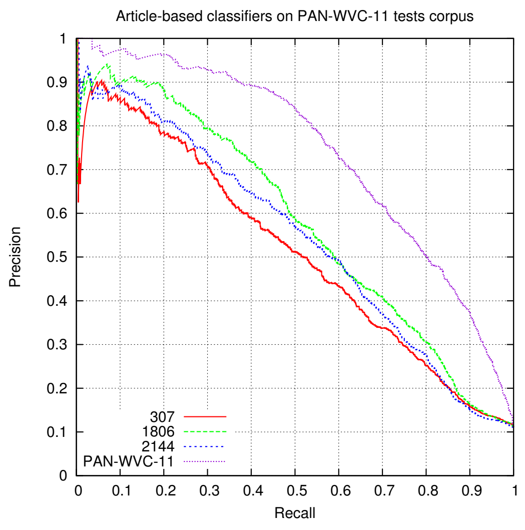
Moreover, we examine the classification performance of a classifier learned on the PAN-WVC-11 training data regarding the article-relative datasets. Finally, for each article, we learn a classifier on its article-relative training dataset and evaluate its performance on the PAN-WVC-11 test dataset. We use a Random Forest classifier with 1000 trees in our experiments. The evaluation results are shown Table 4.2. Due to the high time effort while gathering the simple vandalism datasets, omit features that depend on API requests via a network, such as `EDITS PER USER` and `USER REPUTATION`. The PR curves of the article-relative classifiers in each category in comparison to the PAN-WVC-11-based classifier are given in Figure 4.1. Analyzing the PR-AUC values that are achieved by cross validation, we observe a considerable performance difference using the compiled article-relative datasets. The resulting values range



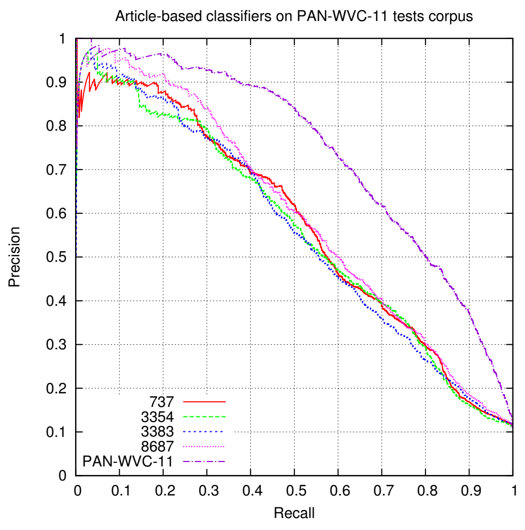
(a) Human Concepts



(b) Fictional Characters



(c) Popular People



(d) Geographical Places

Figure 4.1: PR curves of article-relative classifiers evaluated on PAN-WVC-11. The numbers in the key denote the articles' ids.

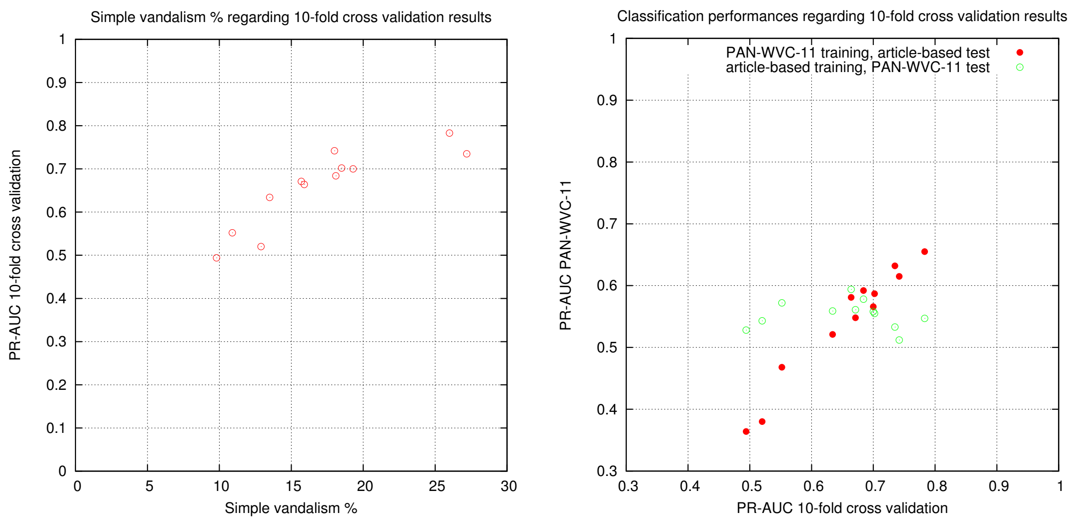


Figure 4.2: Left: PR-AUC values obtained from 10-fold cross validation regarding the percentage of simple vandalism in the article-relative datasets. **Right:** Classification performances (PR-AUC) of the PAN-WVC-11-based classifier on article-relative datasets and of the article-relative classifiers on the PAN-WVC-11 test dataset. Both performance sets are related to the article-relative classifiers' PR-AUC obtained from 10-fold cross validation.

from 0.494 (Anarchism) to 0.783 (Democracy). The evaluation of the PAN-WVC-11-based classifier on the article-relative datasets, as expected, shows the PR-AUC values to correlate with the respective cross validation values of each article-relative dataset (see Figure 4.2). More interesting appears the similar classification performance of all article-relative classifiers on the PAN-WVC-11 test dataset. Furthermore, high percentages of vandalism in the datasets lead to higher cross validation values for the article-relative classifiers.

Similarity of simple vandalism Harpalani et al. [2011] point out different language stylistics for regular and vandalism edits on Wikipedia. Pursuing this approach, we bring up the question, whether vandalism language on Wikipedia differs in various article categories. The idea is to learn an article-relative classifier on data from articles sharing the same category as the article to which the considered edit belongs to. Presuming that the style of vandalism is similar throughout a category, due to a more targeted classifier, the classification performance might increase. In order to proof the basic suitability of such an approach, the similarity of vandalism contents inside of an article category has to be investigated.

As shown in Table 4.3, we calculate a reciprocal classification performance

Table 4.3: Reciprocal classification performance of the article-relative classifiers.

Id	12	5416	7959	4335	8209	307	1806	2144	3354	8687	737	3383	Title
12		0.792	0.699	0.815	0.758	0.729	0.780	0.961	0.807	0.801	0.891	0.840	Anarchism
5416			0.916	0.987	0.975	0.951	0.993	0.830	0.982	0.984	0.892	0.950	Capitalism
7959				0.885	0.940	0.957	0.905	0.703	0.891	0.886	0.808	0.858	Democracy
4335					0.949	0.920	0.978	0.842	0.991	0.992	0.888	0.969	Batman
8209						0.968	0.962	0.786	0.953	0.953	0.857	0.909	Doctor Who
307							0.948	0.759	0.910	0.919	0.862	0.892	Abraham Lincoln
1806								0.811	0.966	0.982	0.884	0.940	A. Schwarzenegger
2144									0.828	0.830	0.930	0.858	Aaliyah
3354										0.995	0.913	0.967	Berlin
8687											0.913	0.974	Detroit
737												0.952	Afghanistan
3383													Brazil
	Anarchism	Capitalism	Democracy	Batman	Doctor Who	Abraham Lincoln	A. Schwarzenegger	Aaliyah	Berlin	Detroit	Afghanistan	Brazil	

measure between two data sets as follows:

$$AUC_{rec} = 1 - |AUC_1 - AUC_2| \tag{4.1}$$

AUC_1 denotes the area under curve (AUC) value of a classifier using article 1 as training dataset with article 2 as test dataset. AUC_2 denotes the reverse case, where training dataset and test dataset are interchanged. The reciprocal classification performance provides a basic measure for the similarity of the properties of reverted content (simple vandalism) in articles 1 and 2. Hence, by combining the performances for the classifiers in a certain category, we are able to asses the similarity of simple vandalism edits between the different chosen categories. Table 4.4 provides an overview of the root mean square (RMS) of AUC_{rec} between categories. Higher values reveal a higher similarity of reverted edit contents, since the vandalism edits of one category can be more accurately detected using data from other categories as training input.

A higher category-internal vandalism similarity is given if the self-similarity (diagonal values) is higher than the similarity regarding other categories. Considering our category set, the category-internal similarities of article-relative datasets in the *Fictional Characters* and *Geographical Places* turn out to be higher than the similarity to all other categories. The category-internal similarities of the *Human Concepts* and the *Popular People* categories are lower

Table 4.4: Simple vandalism similarity between categories using the RMS of reciprocal classification performances of the article-relative classifiers in each category. The diagonal values represent the performance inside of a category. HC = *Human Concepts*, FC = *Fictional Characters*, PP = *Popular People*, GP = *Geographical Places.*

Category	HC	FC	PP	GP
HC	.807	.897	.873	.885
FC		.949	.912	.940
PP			.843	.901
GP				.951

than the similarities to the vandalism contents of all other categories. Since the number of considered articles in each category varies and we did not use categories as they are assigned on Wikipedia, our hypothesis can not be seen to be supported by these results. In future work, this study should be repeated taking a larger amount of Wikipedia articles into account. The used articles should be selected by their categories on Wikipedia and should be equally distributed in each chosen category. Another point of interest might be the impact of the total number of edits and vandalism proportion in an article regarding the observed category-internal similarity.

4.2 Combining Classifiers

The similar classification performance of the aforementioned article-relative classifiers on the PAN-WVC-11 test dataset, as well as their different reciprocal classification performance suggests that each dataset only comprises a part of the vandalism types that are annotated in the PAN-WVC-11 test dataset. Hence, in this section, we attempt to increase the overall classification performance by combining multiple training datasets and classifiers. In a first step, we combine multiple article-relative datasets to learn a classifier. Secondly, we examine the impact of combining multiple article-relative classifiers to build a classification model. Last, we evaluate the combination of a classifier that is learned on the PAN-WVC-11 training dataset combined with partial article-relative datasets.

Combining article-relative datasets Considering the similarity between article-relative datasets, a combination of datasets having a small reciprocal classification performance should lead to an increased classification perfor-

Table 4.5: PR-AUC values of classifiers learned on combined article-relative datasets. The resulting classifiers are evaluate on the PAN-WVC-11 test dataset. The highest PR-AUC values for both sets are marked in **bold**.

	Datasets with highest AUC_{rec} values				
AUC_{rec}	.995	.993	.992	.991	.987
Training datasets	(3354+8687)	(1806+5416)	(4335+8687)	(3354+4335)	(4335+5416)
Single PR-AUC	.561 .594	.578 .555	.558 .594	.561 .558	.558 .555
Combined PR-AUC	**.597**	.572	.585	.569	.558
	Datasets with lowest AUC_{rec} values				
AUC_{rec}	.699	.703	.729	.758	.759
Training datasets	(12+7959)	(2144+7959)	(12+307)	(12+8209)	(307+2144)
Single PR-AUC	.528 .547	.543 .547	.528 .512	.528 .533	.512 .543
Combined PR-AUC	.551	**.572**	.532	.539	.554

Table 4.6: PR-AUC values of the different Vote meta classifiers combining the best performing article-relative classifiers. The classifier is evaluated on the PAN-WVC-11 test dataset. The highest PR-AUC value is marked in **bold**.

Combined classifiers	**2 best**	**3 best**	**4 best**	**4 categories' best**
PR-AUC	.607	**.609**	.609	.602

mance on the PAN-WVC-11 test dataset. We select the five pairs of datasets with the highest, as well as the five pairs of datasets with the lowest reciprocal classification performance. For each pair, we learn a classifier based on the two combined datasets and evaluate all classifiers on the PAN-WVC-11 test dataset. The resulting PR-AUC values are shown in Table 4.5.

We find that in most cases classifiers that are learned on combined article-relative datasets perform better than classifiers learned on the single datasets. For datasets sharing a low similarity of vandalized content, the combined classifiers show a higher performance increase than for datasets sharing a higher similarity.

Combining article-relative classifiers We combine multiple article-relative classifiers by using Weka's `Vote` meta classifier which calculates the total confidence value by a specified voting method. In our experiments we use an average voting of all combined classifiers' confidences. The results of the experiments can be found Table 4.6.

We combine the article-relative classifiers that perform best on the

PAN-WVC-11 test dataset (Detroit (8687), Arnold Schwarzenegger (1806), Afghanistan (373), and Berlin (3354)). Additionally, we build a meta classifier based on the best performing article-relative classifiers of each category (Detroit (8687), Arnold Schwarzenegger (1806), Batman (4335), and Democracy (7959)). We find all combinations to slightly increase the overall classification performance regarding the best performing article-relative classifier which reaches a PR-AUC of 0.594. The best performing classifier combines three article-relative classifiers and achieves a PR-AUC of 0.609.

Combining PAN-WVC-11 and article-relative datasets Using additional vandalism samples might increase the overall classification performance of a basic classifier learned on the PAN-WVC-11 training dataset. In order to create such a semi-supervised training dataset, the original PAN-WVC-11 training dataset can be combined with vandalism samples from the article-relative classifiers. Therefore, we classify the data samples of all article-relative datasets by a classification model which is learned on the PAN-WVC-11 training corpus. We choose the confidence threshold, so that the classifier reaches a precision of 0.964 and 1.0, which leads to a recall of 0.210 and 0.035 on PAN-WVC-11, respectively. From the resulting set of samples that are classified as vandalism, we only use the samples that are found to be simple vandalism edits. This avoids selecting additional false positives from the article-relative datasets. The resulting vandalism datasets comprises 3855 and 402 vandalism samples.

For each vandalism dataset, we compile a combined training dataset by adding its samples to the PAN-WVC-11 training dataset. We train a classifier on each of the combined training data and evaluated it against the PAN-WVC-11 test corpus. Both classifiers do not show an improvement of the classification performance regarding the classifier based on the original training dataset. Evaluating both approaches, the classifier based on the larger additional vandalism dataset reaches a PR-AUC of 0.709, the classifier based on the smaller additional vandalism dataset reaches a PR-AUC of 0.736. We believe that reasons for this performance drop by adding article-relative vandalism edits can be found in the additional vandalism edit's noisy properties regarding regular edits. If the simple vandalism edits on which we compile the additional datasets are false positives and the PAN-WVC-11-based classifier detects them as false positives, then the overall classification performance will drop.

4.3 Conclusions

Based on a Wikipedia article's full revision history, we propose an approach to automatically create article-relative classifiers. These classifiers are based on an article's reverted contents, which can be assumed to be vandalized revisions. We investigate the feasibility of building category-relative classifiers by determining the similarity of vandalism edits between articles and article categories. Using our basic approach to detect simple vandalism and the employed small articles set, we are not able to draw final conclusions on whether the articles inside of a Wikipedia category share similar stylistic properties regarding vandalized contents. We propose to rerun this study on a larger Wikipedia article set and multiple Wikipedia categories.

Using the compiled article-relative datasets, we attempt to improve the classification performance for article-relative classification, as well as for the classifier learned on the PAN-WVC-11 training corpus. The classification performance of a classifier learned on combined article-relative datasets can be increased to a PR-AUC of 0.597. Combining multiple article-relative classifiers by Weka's `Vote` meta classifier leads to an improved classification performance of 0.609 PR-AUC. Augmenting the PAN-WVC-11 training dataset with the most confident vandalism samples from the article-relative datasets results in a lower classification performance.

A problem we are faced with when compiling simple vandalism datasets from Wikipedia's revision history is the rather low precision of our revert extraction approach. In order to improve this process, advanced revert detection algorithms, as proposed by Flöck et al. [2012], should be considered. Suppose that vandalism reverts can be detected with a high accuracy, the found vandalism samples can be used to create unsupervised training sets for vandalism detectors. Thus, one-class classification approaches would fit this kind of problem. This, in turn, requires an adequate classification performance of one-class classification approaches on the Wikipedia vandalism detection problem.

Chapter 5

Conclusions

This chapter summarizes the contributions we make towards detecting vandalism edits on Wikipedia by using machine learning approaches. Finally, possible improvements and future work regarding the presented approaches are pointed out.

5.1 Contributions

We implement a Wikipedia vandalism detection system that can be used to evaluate state-of-the-art classifiers which are learned on custom-built feature sets against the PAN-WVC corpora. Our vandalism detection system is based on the Weka machine learning library and is carried out as a simple-to-use JRuby gem. Considering our chosen feature set, we find Random Forest to be the best performing classifier regarding the PAN-WVC copora.

This thesis contributes to the research of Wikipedia vandalism detection by answering three research questions which are revisited in the subsequent paragraphs.

1. **How does resampling of a skewed training dataset influence Wikipedia vandalism detection performance?**

   The PAN-WVC corpora provide imbalanced Wikipedia vandalism datasets comprising only about 6–7% edits that are annotated as vandalism. Given that training a classifier with an imbalanced dataset often decreases the classification performance, we investigate the impact of training dataset resampling on the classification performance of the Logistic Regression, RealAdaBoost, Random Forest, and Bayesian Network classifiers.

   Applying random undersampling, SMOTE, and a combination of SMOTE and random undersampling, we find that the classification performance of all tested classifiers but Random Forest can be increased by resampling the training dataset. However, Random Forest learned on the original training dataset still outperforms all other tested approaches.

We outline the problems of in-class imbalance and class overlapping to be possible reasons for the poor classification performances that result from using resampling strategies.

2. **What is the performance of a one-class classifier in detecting vandalism on Wikipedia relative to binary classifiers used so far?**

   Since one-class classification approaches have not been applied to the Wikipedia vandalism detection task before, we examine the classification performance of the one-class classifier by Hempstalk et al. [2008] and the one-class SVM by Schölkopf et al. [1999] as implemented by Chang and Lin [2011] on the PAN-WVC corpora.

   Our experimental results show a low classification performance for both one-class classifiers. We find the used one-class classifiers not to be competitive with any of the two-class classifiers used in Chapter 3.

   We believe that reasons for the poor performance can be found in an inappropriate feature set that is used to describe vandalism edits, as well as in inadequate parameter tuning for the used approaches.

3. **Is there an advantage in using a classifier for each article or category? How can an unsupervised or semi-supervised classifier for each article or category be created?**

   Using the full revision histories of Wikipedia articles, in a pilot study, we automatically compile article-relative vandalism datasets which can be used to learn classifiers. In order to automatically compile an article-relative ground truth, we consider reverted edits of an article's revision history to be simple vandalism.

   In this context, we introduce the idea of category-relative classifiers. Such classifiers could take advantage from category-specific vandalism style to predict edits of articles from the same category with a higher performance. In order to estimate the feasibility of category-relative classifiers we investigate the similarity of contents between 12 compiled article-base datasets and four categories to which we assign them.

   Our experimental results show differences in the vandalized contents of two of four analyzed categories. However, since we consider only a small number of articles and categories, it cannot be determined whether the use of category-relative classifiers is a promising approach.

Using the described approaches to compile the article-relative datasets, we find the resulting classifiers not to be competitive with a classifier trained on human-annotated corpora. However, we point out that combining article-relative datasets and article-relative classifiers can improve classification performance regarding single article-relative classifiers.

5.2 Improvements and Future Work

We find Random Forest to be the best performing classifier on the given PAN-WVC corpora. However, there are some drawbacks to using Random Forest, especially when it is employed in real-time applications, such as a Wikipedia vandalism system providing users with an inspection of recently inserted edits. While a larger number of individual trees lead to a higher performance, it also increases the computation time that is needed for classification and particularly for training the classification model. For large training datasets comprising several 10.000 samples (as it is the case for the PAN-WVC corpora and many Wikipedia article based classifiers) this would result in a training time of up to double-digit minutes. A solution might be the Cascaded Random Forest framework developed by Baumann et al. [2013]. Their approach achieves both a speed up model training and an improved classification performance regarding a standard Random Forest classifier.

As shown in Section 3.1.3 training dataset resampling using basic approaches often leads to a lower classification performance. We argue that using advanced dataset cleanup techniques, as proposed by Laurikkala [2001] and Batista et al. [2004], as well as advanced resampling techniques, as proposed by Han et al. [2005] and Maciejewski and Stefanowski [2011], may improve the classification performance. Furthermore, a comprehensive examination of within-class imbalance and class overlapping in the PAN-WVC corpora may give insights into the factors that induce a low or high classification performance. In this context, alternative classification approaches that handle class overlap can be taken into account (Xiong et al. [2013]).

Since our Wikipedia vandalism detection system misses lots of possibly highly decisive features from the literature, the implementation of additional meta data features given in Section 2.4.3 is future work. Moreover, using Lasso as proposed by Javanmardi et al. [2011], the optimal set of features can be determined for a system using the PAN-WVC-11 corpora. Compiling the optimal feature set by using the same methods on one-class classification approaches, we believe that, in combination with parameter tuning, the performance of the aforementioned one-class classification approaches can be improved.

Our approaches of building article-relative classifiers can be improved by using advanced revert detection algorithms (Flöck et al. [2012]). Thus, more accurate annotated article-relative datasets can be build. As future work, the study of similarity of vandalism contents between articles and categories has to be scaled up. A larger amount of articles, as well as categories as they are assigned at Wikipedia, should be considered. Additionally, future work may take into account to use article-relative datasets to learn one-class classifiers, so that automatically compiled vandalism datasets only have to comprise highly accurate vandalism samples.

Appendix A

A.1 Classifier Options

The classifier options we used in the experiments in Chapter 3 and Chapter 4 are listed below.

Table A.1: Weka classifier options used in experiments.

Classifier	Weka Class	Used Options

BayesNet	`weka.classifiers.bayes.BayesNet`	`-D -Q weka.classifiers.bayes.net.search.local.TAN`
		`-- -S BAYES`
		`-E weka.classifiers.bayes.net.estimate.SimpleEstimator`
		`-- -A 0.5`

Logistic Regression	`weka.classifiers.functions.Logistic`	`-R 1.0E-8 -M -1`
Random Forest	`weka.classifiers.trees.RandomForest`	`-I 1000 -K 0 -S 1`

RealAdaBoost	`weka.classifiers.meta.RealAdaBoost`	`-P 100 -H 1.0 -S 1 -I 100`
		`-W weka.classifiers.trees.DecisionStump`

One-class Classifier	`weka.classifiers.meta.OneClassClassifier`	`-num "weka.classifiers.meta.generators.GaussianGenerator`
		`-S 1 -M 0.0 -SD 1.0"`
		`-nom "weka.classifiers.meta.generators.NominalGenerator`
		`-S 1" -trr 0.1 -tcl vandalism -cvr 10 -cvf 10.0 -P 0.5`
		`-S 1 -W weka.classifiers.meta.Bagging`
		`-- -P 100 -S 1 -I 100`
		`-W weka.classifiers.trees.REPTree`
		`-- -M 2 -V 0.001 -N 3 -S 1 -L -1 -I 0.0`

One-class SVM	`weka.classifiers.functions.LibSVM`	`-S 2 -K 2 -D 3 -G 0.0 -R 0.0 -N 0.29 -M 40.0 -C 1.0`
		`-E 0.001 -P 0.1 -Z -seed 1`

A.2 Feature Subsets

A.2.1 Highest Info Gain

We used Weka's information gain attribute selector to determine the top 15 features with highest rankings. The 10 and 15 highest info gain features subset is used in the experiments in Section 3.2. Note that the resulting top 15 feature set is the same for PAN-WVC-10 and PAN-WVC-11.

Table A.2: Top 15 features ranked by info gain. IG denotes **I**nfo **G**ain.

Ranking	PAN-WVC-10		PAN-WVC-11	
	Feature	IG	Feature	IG
1	ALL WORDLISTS FREQUENCY	.0689	ANONYMITY	.0693
2	ANONYMITY	.0610	ALL WORDLISTS FREQUENCY	.0622
3	ALL WORDLISTS IMPACT	.0443	USER REPUTATION	.0585
4	NON-ALPHANUMERIC RATIO	.0438	NON-ALPHANUMERIC RATIO	.0396
5	VULGARISM FREQUENCY	0406	ALL WORDLISTS IMPACT	.0375
6	USER REPUTATION	.0385	VULGARISM FREQUENCY	.0349
7	UPPER TO LOWER CASE RATIO	.0344	WORDS INCREMENT	.0319
8	WORDS INCREMENT	.0320	UPPER TO LOWER CASE RATIO	.0306
9	SIZE INCREMENT	.0291	SIZE INCREMENT	.0287
10	VULGARISM IMPACT	.0290	MARKUP FREQUENCY	.0278
11	MARKUP FREQUENCY	.0260	COMMENT LENGTH	.0269
12	UPPER CASE RATIO	.0253	VULGARISM IMPACT	.0255
13	MARKUP FREQUENCY	.0250	MARKUP IMPACT	.0253
14	COMMENT LENGTH	.0231	UPPER CASE RATIO	.0215
15	SIZE RATIO	.0219	SIZE RATIO	.0209

A.2.2 Features with Non-missing Attributes

Table A.3: Feature sets of features with non-missing values. As non-missing we considered all features with zero or one missing value. Note that the given features apply for PAN-WVC-10. The `TIME INTERVAL` feature (#33) is missing for PAN-WVC-11.

#	Feature
1	ALL WORDLISTS FREQUENCY
2	ARTICLE SIZE
3	BAD FREQUENCY
4	BIASED FREQUENCY
5	BLANKING
6	CHARACTER SEQUENCE
7	CHARACTER DIVERSITY
8	COMPRESSIBILITY
9	EMOTICONS FREQUENCY
10	EMOTICONS IMPACT
11	DIGIT RATIO
12	LONGEST WORD
13	INSERTED EXTERNAL LINKS
14	INSERTED INTERNAL LINKS
15	MARKUP FREQUENCY
16	MARKUP IMPACT
17	NON-ALPHANUMERIC RATIO
18	PRONOUN FREQUENCY
19	REMOVED EMOTICONS FREQUENCY
20	REMOVED MARKUP FREQUENCY
21	REPLACEMENT SIMILARITY
22	SEX FREQUENCY
23	SIZE INCREMENT
24	SIZE RATIO
25	UPPER CASE RATIO
26	UPPER TO LOWER CASE RATIO
27	UPPER CASE WORDS RATIO
28	VULGARISM FREQUENCY
29	WORDS INCREMENT
30	ANONYMITY
31	COMMENT LENGTH
32	EDITS PER USER
33	TIME INTERVAL (only for PAN-WVC-10)
34	TIME OF DAY
35	USER REPUTATION
36	WEEKDAY

Abbreviations

AUC area under curve. 42

ENN Edited Nearest Neighbor Rules. 28

PAN-WVC-10 PAN Wikipedia Vandalism Corpus 2010. 6, 7, 10, 14, 18, 19, 22, 25–28, 31–35, 37, 53, 54

PAN-WVC-11 PAN Wikipedia Vandalism Corpus 2011. 7, 10, 14, 18, 19, 22, 25–28, 31–35, 37, 39, 40, 44–47, 50, 53

PR precision-recall. 8, 32, 34, 40

PR-AUC area under precision-recall curve. 8, 14, 22, 27, 32, 34, 40, 45–47

RMS root mean square. 43

ROC receiver-operator characteristic. 8

ROC-AUC area under receiver-operator characteristic curve. 8

ROS random oversampling. 19

RUS random undersampling. 19, 22, 26, 27, 36

SMOTE **S**ynthetic **M**inority **O**versampling **TE**chnique. 19, 20, 22, 25–28, 34, 36, 48

SVM Support Vector Machine. 31, 34

Webis-WVC-07 Webis Wikipedia Vandalism Corpus 2007. 6, 10

Bibliography

B. Thomas Adler and Luca De Alfaro. A content-driven reputation system for the wikipedia. *Proceedings of the 16th international conference on World Wide Web WWW 07*, 7(Generic):261, 2007. doi: 10.1145/1242572.1242608. URL http://portal.acm.org/citation.cfm?doid=1242572.1242608. 2.4.2

B. Thomas Adler, Luca De Alfaro, and Ian Pye. Detecting wikipedia vandalism using wikitrust. *Notebook Papers of CLEF*, 2010. URL http://institute.lanl.gov/isti/issdm/papers/vandalism-adler-report.pdf. 2.3, 2.1, 2.4.2, 2.4.3, 4.1

B. Thomas Adler, Luca De Alfaro, Santiago M. Mola-Velasco, Paolo Rosso, and Andrew G. West. Wikipedia vandalism detection: Combining natural language, metadata, and reputation features. In *Proceedings of the 12th International Conference on Computational Linguistics and Intelligent Text Processing - Volume Part II*, CICLing'11, pages 277–288, Berlin, Heidelberg, 2011. Springer-Verlag. ISBN 978-3-642-19436-8. URL http://dl.acm.org/citation.cfm?id=1964750.1964776. 2.1, 2.4.2, 2.4.3, 2.5, 2.5, 3.1.2

Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, 6(1):20–29, June 2004. ISSN 1931-0145. doi: 10.1145/1007730.1007735. URL http://doi.acm.org/10.1145/1007730.1007735. 3.1.3, 5.2

Florian Baumann, Arne Ehlers, Karsten Vogt, and Bodo Rosenhahn. Cascaded random forest for fast object detection. In *18th Scandinavian Conference on Image Analysis (SCIA)*, June 2013. 5.2

Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM. ISBN 0-89791-497-X. doi: 10.1145/130385.130401. URL http://doi.acm.org/10.1145/130385.130401. 3.2.3

Leo Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, August 1996. ISSN 0885-6125. doi: 10.1023/A:1018054314350. URL `http://dx.doi.org/10.1023/A:1018054314350`. 3.2.4

Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, October 2001. ISSN 0885-6125. doi: 10.1023/A:1010933404324. URL `http://dx.doi.org/10.1023/A:1010933404324`. 3.1.2

Le S. Cessie and J. C. van Houwelingen. Ridge Estimators in Logistic Regression. *Applied Statistics*, 41(1):191–201, 1992. 3.1.2

Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3): 27:1–27:27, May 2011. ISSN 2157-6904. doi: 10.1145/1961189.1961199. URL `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. (document), 2, 3.3, 2

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16(1):321–357, June 2002. ISSN 1076-9757. URL `http://dl.acm.org/citation.cfm?id=1622407.1622416`. 3.1.1

Si-Chi Chin and W. Nick Street. Enriching wikipedia vandalism taxonomy via subclass discovery. In *LDH*, pages 19–24, 2011. 3.1.3

Si-Chi Chin and W. Nick Street. Divide and transfer: an exploration of segmented transfer to detect wikipedia vandalism. In Isabelle Guyon, Gideon Dror, Vincent Lemaire, Graham W. Taylor, and Daniel L. Silver, editors, *ICML Unsupervised and Transfer Learning*, volume 27 of *JMLR Proceedings*, pages 133–144. JMLR.org, 2012. URL `http://dblp.uni-trier.de/db/journals/jmlr/jmlrp27.html`. 4.1

Si-Chi Chin, W. Nick Street, Padmini Srinivasan, and David Eichmann. Detecting wikipedia vandalism with active learning and statistical language models. In *Proceedings of the 4th Workshop on Information Credibility*, WICOW '10, pages 3–10, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-940-4. doi: 10.1145/1772938.1772942. URL `http://doi.acm.org/10.1145/1772938.1772942`. 4.1

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. In *Proceedings of the 23rd International Conference on Machine Learning*, ICML '06, pages 233–240, New York, NY, USA, 2006. ACM. ISBN 1-59593-383-2. doi: 10.1145/1143844.1143874. URL `http://doi.acm.org/10.1145/1143844.1143874`. 2.3

Misha Denil and Thomas Trappenberg. Overlap versus imbalance. In *Proceedings of the 23rd Canadian Conference on Advances in Artificial Intelligence*, AI'10, pages 220–231, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3-642-13058-5, 978-3-642-13058-8. doi: 10.1007/978-3-642-13059-5_22. URL `http://dx.doi.org/10.1007/978-3-642-13059-5_22`. 3.1.3

Andrew Estabrooks and Nathalie Japkowicz. A mixture-of-experts framework for learning from imbalanced data sets. In *Proceedings of the 4th International Conference on Advances in Intelligent Data Analysis*, IDA '01, pages 34–43, London, UK, UK, 2001. Springer-Verlag. ISBN 3-540-42581-0. URL `http://dl.acm.org/citation.cfm?id=647967.741623`. 3.1.3

Paul F. Evangelista, Mark J. Embrechts, and Boleslaw K. Szymanski. Taming the curse of dimensionality in kernels and novelty detection. In Ajith Abraham, Bernard de Baets, Mario Köppen, and Bertram Nickolay, editors, *Applied Soft Computing Technologies: The Challenge of Complexity*, volume 34 of *Advances in Soft Computing*, pages 425–438. Springer Berlin Heidelberg, 2006. ISBN 978-3-540-31649-7. doi: 10.1007/3-540-31662-0_33. URL `http://dx.doi.org/10.1007/3-540-31662-0_33`. 3.2.4

Fabian Flöck, Denny Vrandečić, and Elena Simperl. Revisiting reverts: Accurate revert detection in wikipedia. In *Proceedings of the 23rd ACM Conference on Hypertext and Social Media*, HT '12, pages 3–12, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1335-3. doi: 10.1145/2309996.2310000. URL `http://doi.acm.org/10.1145/2309996.2310000`. 4.3, 5.2

Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In Lorenza Saitta, editor, *Machine Learning, Proceedings of the Thirteenth International Conference (ICML '96), Bari, Italy, July 3-6, 1996*, pages 148–156. Morgan Kaufmann, 1996. 3.1.2

J. Friedman, T. Hastie, and R. Tibshirani. Additive Logistic Regression: a Statistical View of Boosting. *The Annals of Statistics*, 38(2), 2000. 3.1.2

Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29(2-3):131–163, November 1997. ISSN 0885-6125. doi: 10.1023/A:1007465528199. URL `http://dx.doi.org/10.1023/A:1007465528199`. 3.1.2

Vaishali Ganganwar. An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and ...*, 2(4):42–47, 2012. URL `http://www.ijetae.com/files/Volume2Issue4/IJETAE_0412_07.pdf`. 1, 3

Vicente García, Jose Sánchez, and Ramon Mollineda. An empirical study of the behavior of classifiers on imbalanced and overlapped data sets. In *Proceedings of the Congress on Pattern Recognition 12th Iberoamerican Conference on Progress in Pattern Recognition, Image Analysis and Applications*, CIARP'07, pages 397–406, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-76724-X, 978-3-540-76724-4. URL `http://dl.acm.org/citation.cfm?id=1782914.1782962`. 3.1.3

R. Stuart Geiger and Aaron Halfaker. When the levee breaks: Without bots, what happens to wikipedia's quality control processes? In *Proceedings of the 9th International Symposium on Open Collaboration*, WikiSym '13, pages 6:1–6:6, New York, NY, USA, 2013. ACM. ISBN 978-1-4503-1852-5. doi: 10.1145/2491055.2491061. URL `http://doi.acm.org/10.1145/2491055.2491061`. 1

Arnulf B. A. Graf and Silvio Borer. Normalization in support vector machines. In *Proceedings of the DAGM 2001 Pattern Recognition*, pages 277–282. SpringerVerlag, 2001. 3.2.4

Xinjian Guo, Yilong Yin, Cailing Dong, Gongping Yang, and Guangtong Zhou. On the class imbalance problem. In *Proceedings - 4th International Conference on Natural Computation, ICNC 2008*, volume 4, pages 192–201, 2008. 1

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009. ISSN 1931-0145. doi: 10.1145/1656274.1656278. URL `http://doi.acm.org/10.1145/1656274.1656278`. 2.5

Hui Han, Wen-Yuan Wang, and Bing-Huan Mao. Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *Proceedings of the 2005 International Conference on Advances in Intelligent Computing - Volume Part I*, ICIC'05, pages 878–887, Berlin, Heidelberg, 2005. Springer-Verlag. ISBN 3-540-28226-2, 978-3-540-28226-6. doi: 10.1007/11538059_91. URL `http://dx.doi.org/10.1007/11538059_91`. 3.1.1, 5.2

Manoj Harpalani, Michael Hart, S Signh, Rob Johnson, and Yejin Choi. Language of Vandalism: Improving Wikipedia Vandalism Detection via Stylometric Analysis. *ACL (Short Papers)*, (2009):83–88, 2011. URL `http://www.aclweb.org/anthology/P/P11/P11-2015.pdf`. 2.3, 2.1, 2.4.2, 4.1

Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, September 2009. ISSN 1041-4347. doi: 10.1109/TKDE.2008.239. URL `http://dx.doi.org/10.1109/TKDE.2008.239`. 1, 3, 3.1

Kathryn Hempstalk, Eibe Frank, and Ian H. Witten. One-class classification by combining density and class probability estimation. In *ECML/PKDD (1)*, pages 505–519, 2008. (document), 2, 3.2.2, 3.2.3, 3.2.4, 3.4, 3.4, 3.2.4, 3.2.4, 3.3, 2

Kelly Y. Itakura and Charles L. a. Clarke. Using dynamic markov compression to detect vandalism in the Wikipedia. *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '09*, page 822, 2009. doi: 10.1145/1571941.1572146. URL `http://portal.acm.org/citation.cfm?doid=1571941.1572146`. 2.4.2

Nathalie Japkowicz. Concept-Learning in the Presence of Between-Class and Within-Class Imbalances. In *Canadian Conference on Artificial Intelligence*, pages 67–77, 2001. ISBN 3-540-42144-0. 3.1.3

Sara Javanmardi, David W. McDonald, and Cristina V. Lopes. Vandalism detection in wikipedia: A high-performing, feature-rich model and its reduction through lasso. In *Proceedings of the 7th International Symposium on Wikis and Open Collaboration*, WikiSym '11, pages 82–90, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0909-7. doi: 10.1145/2038558.2038573. URL `http://doi.acm.org/10.1145/2038558.2038573`. 2.3, 2.4.3, 2.4.3, 2.5, 2.5, 5.2

Taeho Jo and Nathalie Japkowicz. Class imbalances versus small disjuncts. *ACM SIGKDD Explorations Newsletter*, 6(1):40–49, June 2004. ISSN 1931-0145. doi: 10.1145/1007730.1007737. URL `http://doi.acm.org/10.1145/1007730.1007737`. 3.1.1, 3.1.3

Shehroz S. Khan and Michael G. Madden. A survey of recent trends in one class classification. In *Proceedings of the 20th Irish Conference on Artificial Intelligence and Cognitive Science*, AICS'09, pages 188–197, Berlin, Heidelberg, 2010. Springer-Verlag. ISBN 3-642-17079-X, 978-3-642-17079-9. URL `http://dl.acm.org/citation.cfm?id=1939047.1939070`. 3.2.2

Aniket Kittur, Bongwon Suh, Bryan A. Pendleton, and Ed H. Chi. He says, she says: conflict and coordination in Wikipedia. In *ACM Conference on Human Factors in Computing Systems*, pages 453 – 462, 2007. ISBN 9781595935939. doi: 10.1145/1240624.1240698. URL `http://portal.acm.org/citation.cfm?id=1240624.1240698`. 1, 2.4.1

Sotiris Kotsiantis, Dimitris Kanellopoulos, and Panayiotis Pintelas. Handling imbalanced datasets : A review. *Science*, 30(1):25–36, 2006. doi: 10.1007/978-0-387-09823-4_45. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.96.9248&amp;rep=rep1&amp;type=pdf`. 1

Jorma Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Proceedings of the 8th Conference on AI in Medicine in Europe: Artificial Intelligence Medicine*, AIME '01, pages 63–66, London, UK, UK, 2001. Springer-Verlag. ISBN 3-540-42294-3. URL `http://dl.acm.org/citation.cfm?id=648155.757340`. 3.1.3, 5.2

T. Maciejewski and J. Stefanowski. Local neighbourhood extension of smote for mining imbalanced data. In *2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, pages 104–111, April 2011. doi: 10.1109/CIDM.2011.5949434. 3.1.1, 5.2

Oleksiy Mazhelis. One-class classifiers : a review and analysis of suitability in the context of mobile-masquerader detection. *South African Computer Journal*, 36:29–48, 2006. URL `http://dblp.uni-trier.de/db/journals/saj/saj36.html`. 3.2.2

David Mease and Abraham Wyner. Evidence contrary to the statistical view of boosting. *The Journal of Machine Learning Research*, 9:131–156, June 2008. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1390681.1390687`. 3.1.2

Santiago Moisés Mola Velasco. Wikipedia vandalism detection through machine learning: Feature review and new proposals - lab report for pan at clef 2010. In *CLEF (Notebook Papers/LABs/Workshops)*, 2010. 2.4.2, 2.1, 2.4.3, 2.4.3, 2.5, 2.5, 2.1, 3.1.2

Judea Pearl and Stuart Russell. Bayesian networks. In M. Arbib, editor, *Handbook of Brain Theory and Neural Networks*, number R-277. MIT Press, 2001. 3.1.2

Martin Potthast. Crowdsourcing a wikipedia vandalism corpus. *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '10*, page 789, 2010. doi: 10.1145/1835449.1835617. URL `http://portal.acm.org/citation.cfm?doid=1835449.1835617`. 1, 2.2, 3.1.3

Martin Potthast and Teresa Holfeld. Overview of the 2nd International Competition on Wikipedia Vandalism Detection. In Vivien Petras, Pamela Forner, and Paul D. Clough, editors, *Notebook Papers of CLEF 11 Labs* *and Workshops*, September 2011. ISBN 978-88-904810-1-7. URL `http://www.clef-initiative.eu/publication/working-notes`. 2.2, 2.4.2, 2.5, 3.1.3

Martin Potthast, Benno Stein, and Robert Gerling. Automatic vandalism detection in wikipedia. In *Advances in Information Retrieval*, pages 663–668. Springer Berlin Heidelberg, 2008. 2.2, 2.4.2, 2.1, 2.4.3, 2.5, 3.1.2

Martin Potthast, Benno Stein, and Teresa Holfeld. Overview of the 1st International Competition on Wikipedia Vandalism Detection. In Martin Braschler, Donna Harman, and Emanuele Pianta, editors, *Working Notes Papers of the CLEF 2010 Evaluation Labs*, September 2010. ISBN 978-88-904810-2-4. URL `http://www.clef-initiative.eu/publication/working-notes`. 2.2, 2.4.2

Bernhard Schölkopf, Robert Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support Vector Method for Novelty Detection. In *Advances in Neural Information Processing Systems 12*, pages 582–588, 1999. ISBN 0-262-11245-0. 3.2.2, 3.2.3, 3.3, 2

Mark R. Segal. Machine learning benchmarks and random forest regression. *Center for Bioinformatics & Molecular Biostatistics*, 2004. 3.1.2

Koen Smets, Bart Goethals, and Brigitte Verdonk. Automatic vandalism detection in wikipedia: Towards a machine learning approach. In *In WikiAI '08: Proceedings of the AAAI Workshop on Wikipedia and Artificial Intelligence*, 2008. 2.4.2, 2.1, 3.1.2

Alexander R. Statnikov, Lily Wang, and Constantin F. Aliferis. A comprehensive comparison of random forests and support vector machines for microarray-based cancer classification. *BMC Bioinformatics*, 9, 2008. URL `http://dblp.uni-trier.de/db/journals/bmcbi/bmcbi9.html`. 3.1.2

David Martinus Johannes Tax. *One-class classification: Concept learning in the absence of counter-examples.* PhD thesis, Technische Universiteit Delft, 2001. URL `http://proquest.umi.com/pqdweb?did=728104171&Fmt=2&clientId=36097&RQT=309&VName=PQD`. 3.2.1, 3.2.2

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994. 2.5

Ivan Tomek. Two modifications of CNN. *IEEE Transactions on Systems Man and Communications, SMC*, 6:769–772, 1976. 3.1.3

Khoi-Nguyen Tran and Peter Christen. Cross Language Prediction of Vandalism on Wikipedia Using Article Views and Revisions. *Advances in Knowledge Discovery and Data Mining*, pages 268–279, 2013. URL `http://link.springer.com/chapter/10.1007/978-3-642-37456-2_23`. 1, 2.1, 2.4.2

Jason Van Hulse, Taghi M. Khoshgoftaar, and Amri Napolitano. Experimental perspectives on learning from imbalanced data. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, pages 935–942, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-793-3. doi: 10.1145/1273496.1273614. URL `http://doi.acm.org/10.1145/1273496.1273614`. 3.1.3, 3.3, 3.3

William Yang Wang and Kathleen R. McKeown. "Got You!": Automatic vandalism detection in Wikipedia with web-based shallow syntactic-semantic modeling. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 1146–1154, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics. URL `http://dl.acm.org/citation.cfm?id=1873781.1873910`. 2.1, 2.4.3, 2.5

Andrew G. West and Insup Lee. Multilingual vandalism detection using language-independent & ex post facto evidence - notebook for pan at clef 2011. In *CLEF (Notebook Papers/Labs/Workshop)*, 2011. 2.1, 2.4.2, 2.4.3, 2.5, 2.5, 2.5, 2.1

Andrew G. West, Sampath Kannan, and Insup Lee. Detecting wikipedia vandalism via spatio-temporal analysis of revision metadata. In *Proceedings of the Third European Workshop on System Security*, EUROSEC '10, pages 22–28, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0059-9. doi: 10.1145/1752046.1752050. URL `http://doi.acm.org/10.1145/1752046.1752050`. 2.4.2, 2.4.3, 4.1, 4.1

Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, 2(3):408–421, 1972. URL `http://dblp.uni-trier.de/db/journals/tsmc/tsmc2.html`. 3.1.3

Haitao Xiong, Tongqiang Jiang, and Shouxiang Zhao. Ensemble Learning Method for Class Overlapping Problem. *Journal of Information & Computational Science*, 10(4):1195–1202, 2013. 5.2