Ageing-Based Multinomial Naive Bayes Classifiers Over Opinionated Data Streams

Sebastian Wagner[1], Max Zimmermann[2], Eirini Ntoutsi[1(✉)], and Myra Spiliopoulou[2]

[1] Ludwig-Maximilians University of Munich (LMU), Munich, Germany
sebastian.t.wagner@campus.lmu.de, ntoutsi@dbs.ifi.lmu.de
[2] Otto-von-Guericke-University Magdeburg, Magdeburg, Germany
{max.zimmermann,myra}@iti.cs.uni-magdeburg.de

Abstract. The long-term analysis of opinionated streams requires algorithms that predict the polarity of opinionated documents, while adapting to different forms of concept drift: the class distribution may change but also the vocabulary used by the document authors may change. One of the key properties of a stream classifier is adaptation to concept drifts and shifts; this is typically achieved through ageing of the data. Surprisingly, for one of the most popular classifiers, Multinomial Naive Bayes (MNB), no ageing has been considered thus far. MNB is particularly appropriate for opinionated streams, because it allows the seamless adjustment of word probabilities, as new words appear for the first time. However, to adapt properly to drift, MNB must also be extended to take the age of documents and words into account.

In this study, we incorporate ageing into the learning process of MNB, by introducing the notion of *fading* for words, on the basis of the recency of the documents containing them. We propose two fading versions, gradual fading and aggressive fading, of which the latter discards old data at a faster pace. Our experiments with Twitter data show that the ageing based MNBs outperform the standard accumulative MNB approach and manage to recover very fast in times of change. We experiment with different data granularities in the stream and different data ageing degrees and we show how they "work together" towards adaptation to change.

1 Introduction

Nowadays, we experience an increasing interest on word-of-mouth communication in social media, including opinion sharing [16]. A vast amount of voluntary and bona fide feedback accumulates, referring to products, persons, events etc. Opinionated information is valuable for consumers, who benefit from the experiences of other consumers, in order to make better buying decisions [13], but also for vendors, who can get insights on what customers like and dislike [18]. The extraction of such insights requires a proper analysis of the opinionated data.

In this work, we address the issue of polarity learning over opinionated streams. The accumulating opinionated documents are subject to different forms of drift: the subjects discussed change, the attitude of people towards specific

A. Appice et al. (Eds.): ECML PKDD 2015, Part I, LNAI 9284, pp. 401–416, 2015.
DOI: 10.1007/978-3-319-23528-8_25

products, events or other forms of entities change, the vocabulary changes. As a matter of fact, the impact of the vocabulary is less investigated. It is well-known that the polarity of some words depends on the context they are used in; this subject is investigated e.g. in the context of recurring concepts [8]. However, the impact of the vocabulary in an opinionated stream is much more broad: an opinionated word can appear in more contexts than we can (or want to) trace, since some contexts are rare or do not recur. More importantly, new words emerge and some words are used less. This implies that the polarity learner should be able to cope with an evolving feature space. To deal with concept drift in the opinionated data and their feature space, we propose a *fading* Multinomial Naive Bayes polarity learner. We extend the Multinomial Naive Bayes (MNB) with an ageing function that gradually forgets (fades away) old data and outdated words.

Multinomial Naive Bayes (MNB) classifiers comprise one of the most well-known classifiers and are widely used also for sentiment analysis although most of the approaches cover the non-stream case [17]. For opinionated streams, MNB has a *cardinal advantage*: it allows the seamless adaptation of the vocabulary, by simply requiring the computation of the class probabilities for each word. No other stream classification algorithm can respond so intuitively to an evolving feature space. Surprisingly, so far, MNB is used in opinionated streams mostly without forgetting past data and without extending to a new vocabulary [1]. This is problematic, because the model may overfit to old data and, fore-mostly, to an outdated vocabulary. In this study, we propose two different forgetting mechanisms that differ on how drastically they forget over time.

The rest of the paper is organized as follows: Related work is discussed in Section 2. The basic concepts and motivation are presented in Section 3. The ageing-based MNBs are introduced in Section 4. Experimental results are shown in Section 5. Conclusions and open issues are discussed in Section 6.

2 Related Work

Stream mining algorithms typically assume that the most recent data are the most informative, and thus employ different strategies to downgrade old, obsolete data. In a recent survey [6], Gama et al. discuss two forgetting mechanisms: (i) *abrupt forgetting* where only recent instances, within a sliding window, contribute to the model, and (ii) *gradual forgetting* where all instances contribute to the model but with a weight that is regulated by their age. In the context of our study, the forgetting strategy also affects the vocabulary – the feature space. In particular, if a set of documents is deleted (abrupt forgetting), all words that are in them but in none of the more recent documents are also removed from the feature space. This may harm the classifier, because such words may re-appear soon after their removal. Therefore, we opt for gradual forgetting.

Another approach for selecting features/ words for polarity classification in a dynamic environment is presented in [11]. In this approach, selection does not

[1] An exception is our own prior work [24,25].

rely on the age of the words but rather on their *usefulness*, defined as their contribution to the classification task. Usefulness is used in [11] as a selection criterion, when the data volume is high, but is also appropriate for streams with recurring concepts. Concept recurrence is studied e.g. in [8] (where meta-classifiers are trained on data referring to a given concept), and in [12] (where a concept is represented by a data bucket, and recurrence refers to similar buckets). Such methods can be beneficial for opinion stream classification, *if* all encountered opinionated words can be linked to a reasonably small number of concepts that do recur. In our study, we do not pose this requirement; word polarity can be assessed in our MNB model, without linking the words to concepts.

Multinomial Naive Bayes (MNB) [14] is a popular classifier due to its simplicity and good performance, despite its assumption on the class-conditional independence of the words [5,22]. Its simplicity and easy online maintenance constitutes it particularly appealing for data streams. As pointed out in Section 1, MNB is particularly appropriate for adaptation to an evolving vocabulary. In [24,25], we present functions that recompute the class probabilities of each word. In this study, we use different functions, as explained in the last paragraph of this section.

Bermingham et al. [2] compared the performance of Support Vector Machines (SVM) and MNB classifiers on microblog data and reviews (not streams) and showed that MNB performs well on short-length, opinion-rich microblog messages (rather than on long texts). In [10], popular classification algorithms were studied such as MNBs, Random Forest, Bayesian Logistic Regression and SVMs using sequential minimal optimization for the classification in Twitter streams while building classifiers at different samples. Across the tested classifiers, MNBs showed the best performance for all applied data sets.

In [3], MNB has been compared to Stochastic Gradient Descend (SGD) and Hoeffding Trees for polarity classification on streams. Their MNB approach is incremental, i.e., it accumulates information on class appearances and word-in-class appearances over the stream, however, it did not forget anything. Their experiments showed that MNB had the largest difficulty in dealing with drifts in the stream population, although its performance in times of stability was very good. Regarding runtime, MNB was the fastest model due to its simplicity in predictions but also due to the easy incorporation of new instances in the model. The poor performance of MNB in times of change was also observed in [21], and triggered our ageing-based MNB approach.

Closest to our approach is our earlier work [24]. There, MNB is in the core of a polarity learner that uses two adaptation techniques: i) a forward adaptation technique that selects "useful" instances from the stream for model update and ii) a backward adaptation technique that downgrades the importance of old words from the model based on their age. There are two differences between that earlier method (and our methods that build upon it, e.g. [25]) and the work proposed here. First, the method in [24] is semi-supervised: after receiving an initial seed of labeled documents, it relies solely on the labels it derives from the learner. Backward adaptation is not performing well in that scenario,

presumably because the importance of the words in the initial seed of documents diminishes over time. Furthermore, in [24], the ageing of the word-class counts is based directly upon the age of the original documents containing the words. The word-class counts are weighted locally, i.e., within the documents containing the words, and the final word-class counts are aggregations of these local scores. In our current work, we do not monitor the age of the documents. Rather, we use the words as first class objects, which age with time. Therefore the ageing of a word depends solely on the last time the word has been observed in some document from the stream.

3 Basic Concepts

We observe a stream $\mathcal{S}$ of opinionated documents arriving at distinct timepoints $t_0, \ldots, t_i, \ldots$; at each t_i a batch of documents might arrive. The definition of the batch depends on the application per se: i) one can define the batch at the instance level, i.e., a fixed number of instances is received at each timepoint or ii) at the temporal level, i.e., the batch consists of the instances arriving within each time period, e.g. on a daily basis if day is the considered temporal granularity. A document $d \in \mathcal{S}$ is represented by the *bag-of-words* model and for each word $w_i \in d$ its frequency f_i^d is also stored.

Our goal is to build a polarity classifier for the prediction of the polarity of new arriving documents. As it is typical in streams, the underlying population might undergo changes over time, referred in the literature as *concept drift*. The changes are caused by two reasons: i) change in the sentiment of existing words (for example, words have different sentiment for different contexts, e.g. the word "heavy" is negative for a camera, but positive for a solid wood piece of furniture); ii) new words might appear over time and old words might become obsolete (for example, new topics emerge all the time in the news and some topics are not mentioned anymore). The drift in the population might be gradual or drastic; the later is referred also as *concept shift* in the literature. A stream classifier should be able to adapt to drift while maintaining a good predictive power. Except for the *quality of predictions*, another important factor for a stream classifier is *fast adaptation* to the underlying evolving stream population.

3.1 Basic Model: Multinomial Naive Bayes

According to the underpinnings of the Multinomial Naive Bayes (MNB) [14], the probability of a document d belonging to a class c is given by:

$$P(c|d) = P(c) \prod_{i=1}^{|d|} P(w_i|c)^{f_i^d} \tag{1}$$

where $P(c)$ is the prior probability of class c, $P(w_i|c)$ is the conditional probability that word w_i belongs to class c and f_i^d is the number of occurrences of w_i in document d. These probabilities are typically estimated based on a dataset

$\mathcal{D}$ with class labels (training set); we indicate the estimates from now on by a "hat" as in $\hat{P}$.

The class prior $P(c)$ is easily estimated as the fraction of the set of training documents belonging to class c, i.e.,:

$$\hat{P}(c) = \frac{N_c}{|\mathcal{D}|} \tag{2}$$

where N_c is the number of documents in $\mathcal{D}$ belonging to class c and $|\mathcal{D}|$ is the total number of documents in $\mathcal{D}$.

The conditional probability $P(w_i|c)$ is estimated by the relative frequency of the word $w_i \in V$ in documents of class c:

$$\hat{P}(w_i|c) = \frac{N_{ic}}{\sum_{j=1}^{|V|} N_{jc}} \tag{3}$$

where N_{ic} is the number of occurrences of word w_i in documents with label c in $\mathcal{D}$, V is the vocabulary over the training set $\mathcal{D}$. For words that are unknown during prediction, i.e., not in V, we apply the Laplace correction and initialize their probability to $1/|V|$.

From the above formulas, it is clear that the quantities we need in order to estimate the class prior $\hat{P}(c)$ and the class conditional word estimates $\hat{P}(w_i|c)$ are the class prior counts N_c and the word-class counts N_{ic}, where $w_i \in V, c \in C$ are all computed from the training set $\mathcal{D}$. The conventional, static MNB uses the whole training set $\mathcal{D}$ at once to compute these counts.

The typical extension for data streams [3], updates the counts based on new instances from the stream. Let d be a new incoming document from the stream $\mathcal{S}$ with class label c. Updating the MNB model means actually updating the class and word-class counts based on the incoming document. In particular, the number of documents belonging to class c is increased, i.e.,: $N_c+ = 1$. Similarly, the class-word counts for each word $w_i \in d$ are updated, i.e., $N_{ic}+ = f_i^d$, where f_i^d is the frequency of w_i in d. For existing words in the model, this implies just an update of their counts. For so-far unknown words, this implies that a new entry is created for them in the model. These accumulative counts are used during polarity prediction in Equations 2 and 3. We refer to this model as *accumulativeMNB*.

From the above description it is clear that the typical MNB stream model exhibits a very *long memory* as nothing is forgotten with time, and therefore it cannot respond fast to changes. Next, we present our extensions to MNB: we weight documents and words on their age, ensuring that the model forgets and responds faster to change.

4 Ageing-Based Multinomial Naive Bayes

The reason for the poor performance of an MNB model in times of change is its accumulative nature. As already mentioned, once entering the model nothing is

forgotten, neither words nor class prior information. To make MNBs adaptable to change, we introduce the notion of time (Section 4.1) and we show how such a "temporal" model can be used for polarity learning in a stream environment (Sections 4.2, 4.3).

4.1 Ageing-Based MNB Model

In the typical, accumulative MNB classifier there is no notion of time, rather all words are considered equally important independently of their arrival times. We couple the MNB with an ageing mechanism that allows for a differentiation of the words based on their last observation time in the stream.

Ageing is one of the core mechanism in data streams for dealing with concept drifts and shifts in the underlying population. Several ageing mechanisms have been proposed [7] including the landmark window model, the sliding window model and the damped window model. We opt for the damped window model, as already explained, as it comprises a natural choice for temporal applications and data streams [1,4,15]. According to the *damped window model*, the data are subject to ageing based on an ageing function so that recent data are considered more important than older ones. One of the most commonly used ageing functions is the *exponential fading function* that exponentially decays the weights of data instances with time.

Definition 1 (Ageing function). *Let d be a document arriving from the stream $\mathcal{S}$ at timepoint t_d. The weight of d at the current timepoint $t \geq t_d$ is given by: $age(d,t) = e^{-\lambda \cdot (t-t_d)}$ where $\lambda > 0$ is the decay rate.*

The weight of the document d decays over time based on the time period elapsing from the arrival of d and the decay factor λ. The higher the value of λ, the lower the impact of historical data comparing to recent data. The ageing factor λ is critical for the ageing process as it determines what is the contribution of old data to the model and how fast old data is forgotten. Another way of thinking of λ is by considering that $\frac{1}{\lambda}$ is the period for an instance to loose half of its original weight. For example, if $\lambda = 0.5$ and timestamps correspond to days, this means that $\frac{1}{0.5} = 2$ days after its observation an instance will loose 50% of its weight. For $\lambda = 0$ there is no ageing and therefore the classifier is equivalent to the accumulative MNB (cf. Section 3.1).

The timestamp of the document from the stream is "transferred" to its component words and finally, to the MNB model. In particular, each word-class pair (w, c) entry in the model is associated with

- *the last observation time*, t_{lo}, which represents the last time that the word w has been observed in the stream in a document of class c.

The t_{lo} entry indicates how recent is the last observation of word w in class c.

Similarly, each class entry in the model is associated with a last observation timestamp indicating the last time that the class was observed in the stream.

Based on the above, the ageing-based MNB model consists of the temporally annotated class prior counts (N_c, t^c_{lo}) and class conditional word counts (N_{ic}, t^{ic}_{lo}).

Hereafter, we focus on how such a temporal model can be used for prediction while being maintained online. We distinguish between a normal fading MNB approach (Section 4.2) and an aggressive/drastic fading MNB approach (Section 4.3).

4.2 Ageing-Based MNB Classification

In order to *predict* the class label of a new document d arriving from the stream $\mathcal{S}$ at timepoint t, we employ the ageing-based version of MNB: in particular, the temporal information associated with the class prior counts and the class conditional word counts is incorporated in the class prior estimation $\hat{P}(c)$ and in the class conditional word probability estimation $\hat{P}(w_i|c)$, i.e., in Equations 2 and 3, respectively.

The updated temporal class prior for class $c \in C$ at timepoint t is given by :

$$\hat{P}^t(c) = \frac{N^t_c * e^{-\lambda \cdot (t - t^c_{lo})}}{|\mathcal{S}^t|} \tag{4}$$

where N^t_c is the number of documents in the stream up to timepoint t belonging to class c and $|\mathcal{S}^t|$ is the total number of document in the stream thus far, which can be easily derived from the class counts as $|\mathcal{S}^t| = \sum_{c' \in C} N^t_{c'}$. Note that the class counts, N_c, are maintained online over the stream as described below. The t^c_{lo} is the last observation of class c in the stream and $(t - t^c_{lo})$ describes the temporal gap from the last appearance of the class label c in the stream to timepoint t.

The updated temporal class conditional word probability for a word $w_i \in d$ at t is given by:

$$\hat{P}^t(w_i|c) = \frac{N^t_{ic} * e^{-\lambda \cdot (t - t^{(w_i,c)}_{lo})}}{\sum_{j=1}^{|V^t|} N^t_{jc} * e^{-\lambda \cdot (t - t^{(w_j,c)}_{lo})}} \tag{5}$$

where N^t_{ic} is the number of appearances of word w_i in class c in the stream up to timepoint t and V^t is the vocabulary (i.e., distinct words) accumulated from the stream up to timepoint t. It is stressed that the vocabulary changes over time as new words may arrive from the stream.

The word conditional class counts, N_{ic}, are also maintained online over the stream as described hereafter.

Online Model Maintenance. The update of the MNB model consists of updating the class count and word-class count entries and their temporal counterparts. If d with timestamp t is a new document to be included in the MNB model and c is its class label, then the corresponding class count is increased by one, i.e., $N_c += 1$ and for each word $w_i \in d$, the class conditional word counts

N_{ic} are increased based on the frequency of w_i in d, i.e., $N_{ic} += f^d_{w_i}$. The temporal counterpart of N_c is updated w.r.t. arrival time t of d, i.e., $t^c_{lo} = t$.

Similarly the temporal counterparts of any word class combination count N_{ic} in d will be updated, i.e., $t^{ic}_{lo} = t$. We refer to this method as *fadingMNB* hereafter.

4.3 Aggressive Fading MNB Alternative

The *fadingMNB* approach presented in the previous subsection, accumulates evidence about class appearances and word-class appearances in the stream up to the current time and applies the ageing function upon these accumulated counts. The bigger the gap between the last observation of a word in a class and the current timepoint, the more the weight of this word in the specific class would be decayed. However, as soon as we observe the word-class combination again in the stream, the total count is revived. This is because the counts are accumulative and the ageing function is applied *a posteriori*.

An alternative approach to make the exponential fading even more rapid and adapt more quickly to changes is to store in the model the aged-counts instead of the accumulated ones. That is, the ageing is applied over the faded counts. Obviously, such an approach implies a more drastic ageing of the data compared to *fadingMNB*. The decay is not exponential anymore and it depends also on how often a word is observed in the stream. In particular, words that appear constantly in the stream, i.e., at each time point, will not be affected (as with the *fadingMNB* approach) but if some period intervenes between consecutive appearances of a word, the word will be "penalized" for this gap. We refer to this method as *aggressiveFadingMNB*.

5 Experiments

In our experiments, we compare the original MNB stream model (*accumulativeMNB*), to our fading MNB model (*fadingMNB*) and to the aggressive fading MNB model (*aggressiveFadingMNB*). In Section 5.1, we present the Twitter dataset we use for the evaluation, and in Section 5.2 we present the evaluation measures. We present the results on classfication quality in Section 5.3. In Section 5.4, we discuss the role of the fading factor λ. We have run these experiments on different time granularities: hour, day and week. Due to lack of space, we report only on the *hourly-aggregated stream* in Sections 5.3 and 5.4. Then, in Section 5.5, we discuss the role of the stream granularity on the quality of the different models and how it is connected to the fading factor λ.

5.1 Data and Concept Changes

We use the TwitterSentiment dataset [19], introduced in [9]. The dataset was collected by querying the Twitter API for messages between April 6, 2009 and

June 25, 2009. The query terms belong to different categories, such as companies (e.g. query terms "aig", "at&t"), products (e.g. query terms "kindle2", "visa card"), persons (e.g. "warren buffet", "obama"), events (e.g. "indian election", "world cup"). Evidently, the stream is very heterogeneous. It has not been labeled manually. Rather, the authors of [9] derived the labels with a Maximum Entropy classifier that was trained on emoticons.

We preprocessed the dataset as in our previous work [21], including following steps: (i) dealing with data negation (e.g. replacing "not good" with "not_good", "not pretty" with "ugly" etc.), (ii) dealing with colloquial language (e.g. converting "luv" to "love" and "youuuuuuuuu" to "you"), (iii) elimination of superfluous words (e.g. Twitter signs like or #), stopwords (e.g. "the", "and"), special characters and numbers, (iv) stemming (e.g. "fishing", "fisher" were mapped to their root word "fish"). A detailed description of the preprocessing steps is in [20].

The final stream consists of 1,600,000 opinionated tweets, 50% of which are positive and 50% negative (two classes). The class distribution changes over time.

How to choose the temporal granularity of such a stream? On Figure 1, we show the tweets at different levels of temporal granularity: weeks (left), days (center), hours (right). The *weekly-aggregated stream* (Figure 1, left) consists of #12 distinct weeks (the x-axis shows the week of the year). Both classes are present up to week 25, but after that only instances of the negative class appear. In the middle of the figure, we see the same data aggregated at day level: there are #49 days (the horizontal axis denotes the day of the year). Up to day 168, we see positive and negative documents; the positive class (green) is overrepresented. But towards the end of the stream the class distribution changes and the positive class disappears. We see a similar behavior in the *hourly-aggregated stream* (Figure 1, right), where the x-axis depicts the hour (of the year). On Figure 1, we see that independently of the aggregation level, the amount of data received at each timepoint varies: there are high-traffic time points, like day 157 or week 23 and low-traffic ones, like day 96 or week 15. Also, there are "gaps" in the monitoring period. For example, in the daily-aggregated stream there are several 1-day gaps like day 132 but also "bigger gaps" like 10 days of missing observations between day 97 and day 107.

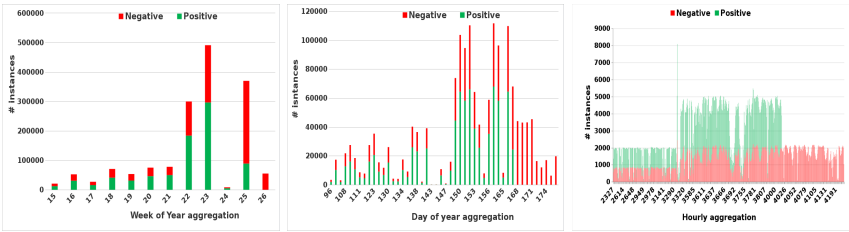

Fig. 1. Class distribution in the stream at different temporal granularities: weekly (left), daily (center), hourly (right).

The time granularity affects the ageing mechanism, since all documents associated with the same time unit (e.g. day) have the same age/weight. In the following, we experiment with all three levels of granularity.

5.2 Evaluation Methods and Evaluation Measures

The two most popular methods for evaluating classification algorithms are hold-out evaluation and prequential evaluation. Their fundamental difference is in the order in which they perform training and testing and the ordering of the dataset [7]. In *hold-out evaluation*, the current model is evaluated over a single independent hold-out test set. The hold-out set is the same over the whole course of the stream. For our experiments, the hold-out set consists of 30% of all instances randomly selected from the stream. In *prequential evaluation*, each instance from the stream is first used for testing and then for training the model. This way, the model is updated continuously based on new instances.

To evaluate the quality of the different classifiers, we employed accuracy and kappa [23] over an *evaluation window*, *evalW*. *Accuracy* is the percentage of correct classifications in w. Bifet et al. [3] use the kappa statistic defined as $k = \frac{p_0 - p_c}{1 - p_c}$, where p_0 is the accuracy of the studied classifier and p_c is the accuracy of the chance classifier. Kappa lies between -1 and 1.

5.3 Classifier Performance

Accuracy and Kappa in Prequential Evaluation. We compare the performance of our *fadingMNB* and *aggressiveFadingMNB* to the original *accumulativeMNB* algorithm in the hourly-aggregated stream, using prequential evaluation. As criteria for classification performance, we use accuracy (Figure 2) and kappa (Figure 3). In both figures, we see that classification performance has two phases, before and after the arrival of instance 1,341,000; around that time, there has been a drastic change in the class distribution, cf. Figure 1. We discuss these two phases, i.e., the left, respectively right part of the figures, separately.

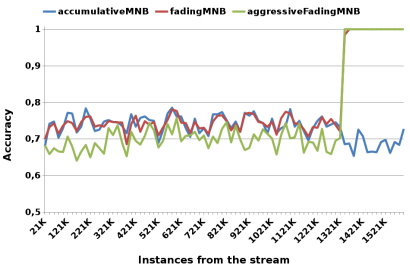

Fig. 2. Prequential evaluation on *accuracy* in the hourly-aggregated stream – fading factor $\lambda = 0.1$, evaluation window $evalW = 1000$

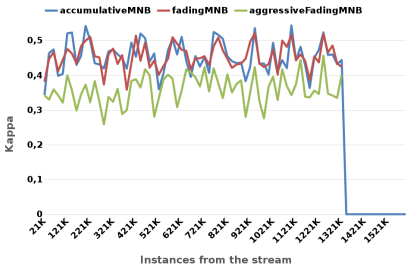

Fig. 3. Prequential evaluation on *kappa* in the hourly-aggregated stream – same parameters as in Figure 2, i.e., $\lambda = 0.1$, $evalW = 1000$

The left part of the accuracy plots on Figure 2 shows that *accumulativeMNB* has the best accuracy, followed closely by *fadingMNB*, while *aggressiveFadingMNB* has slightly inferior performance. In the left part of the plots on kappa (Figure 3), the relative performance is the same, but the performance inferiority of *accumulativeMNB* is more apparent.

In the right part of the accuracy plots on Figure 2, i.e., after the drastic change in the class distribution, we see that *accumulativeMNB* experiences a slight performance drop, while the accuracy of our two algorithms ascends rapidly to 100% (the two curves coincide). The intuitive explanation for the inferior performance of *accumulativeMNB* is that it remembers all past data, so it cannot adapt to the disappearance of the positive class. The proposed *fadingMNB* and *aggressiveFadingMNB*, on the contrary, manage to recover after the change.

The right part of the plots on kappa (Figure 3) gives a different picture: the performance of all three algorithms drops to zero after the concept shift (the three curves coincide). This is owed to the nature of kappa: it juxtaposes the performance of the classifier to that of a random classifier; as soon as there is only one class in the data (here: the negative one), no classifier can be better than the random classifier. Since the accuracy plots and the kappa plots show the same trends before the drastic concept change, and since the accuracy plots reflect the behavior of the classifiers after the change much better than kappa does, we concentrate on accuracy as evaluation measure hereafter.

Accuracy in Hold-Out Evaluation. Under prequential evaluation, each labeled document is used first for testing and, then, immediately for learning. In a more realistic setting, we would expect that the "expert" is not available all the time to deliver fresh labels for each incoming document from the stream. We are therefore interested to study the performance of the algorithms when less labeled data can be exploited. In this experiment, we train the algorithms in a random sample of 70% of the data and test them in the remaining 30%. The results are on Figure 4.

As pointed out in Figure 4, this hold-out evaluation was done after learning on 70% of the complete stream. Thus, the concept drift is incorporated into the learned model and the performance is stable. This allows us to highlight the influence of the hold-out data on the vocabulary. In particular, since the test instances constitute 30% of the dataset, some words belonging to them may be absent from the vocabulary used for training, or be so rare in the training sample that the class probabilities have not been well estimated. This effect cannot be traced under prequential evaluation, because all instances are gradually incorporated into the model. With our variant of a hold-out evaluation, we can observe on Figure 4 how the absence of a part of the vocabulary affects performance.

Figure 4 shows that *aggressiveFadingMNB* performs very poorly. This is expected, because this algorithm forgets instances too fast and thus cannot maintain good estimates of the polarity probabilities of the words. More remarkable is the performance difference between *fadingMNB* and *accumulativeMNB*: the gradual fading of some instances leads to a better model! An explanation may

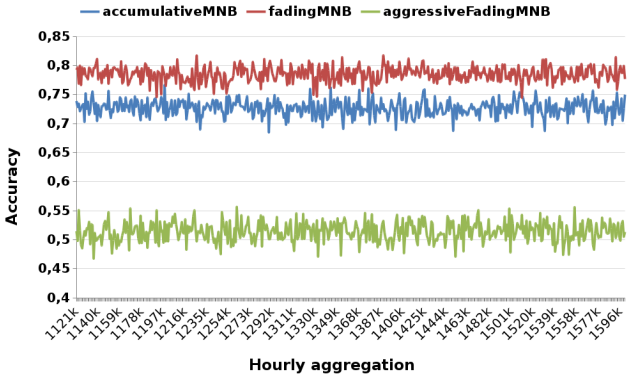

Fig. 4. Hold-out evaluation on *accuracy* in the hourly-aggregated stream – same parameters as in Figure 2, i.e., $\lambda = 0.1$, $evalW = 1000$; 30% of the data are used for testing the stream, after learning on the *complete* training sample (70% of the data).

be that *accumulativeMNB* experiences an overfitting on the large vocabulary of *all* training instances, while *fadingMNB* forgets some training data and thus uses a "smaller vocabulary" that is still adequate to predict the labels of the test instances. We intend to investigate this further by studying the contents of the vocabularies used by the learning algorithms.

5.4 Impact of the Fading Factor λ on the New Algorithms

On Figure 2, we have seen that both *fadingMNB* and *aggressiveFadingMNB* adapted immediately to the drastic change in the class distribution. In this experiment, we investigate how the fading factor λ affects the accuracy of the classifiers. First, we compare the performance of the two algorithms for a very small value of λ, cf. Figure 5.

Figure 5 shows that *aggressiveFadingMNB* manages to adapt to changes, while *fadingMNB* does not. Since small λ values increase the impact of the old data, i.e., enforce a *long memory*, the performance of *fadingMNB* deteriorates, as is the case for *accumulativeMNB*. In contrast, *aggressiveFadingMNB* needs such a small λ to remember some data in the first place.

As an extreme case, $\lambda = 0$ implies that no forgetting takes place and therefore corresponds to *accumulativeMNB*. A high value of λ implies that *aggressiveFadingMNB* forgets all data; a very low value of λ causes *fadingMNB* to remember a lot and therefore it degenerates to *accumulativeMNB*. To visualize these "connections" and to better understand the effect of the fading factor λ on each method, we experiment with different values of λ over the hourly-aggregated stream, cf. Figure 6.

Figure 6 depicts a constant performance of *accumulativeMNB* as it does not depend on λ. The *fadingMNB* and the *aggressiveFadingMNB* have a complementary performance. For small values of λ, *aggressiveFadingMNB* performs best; as

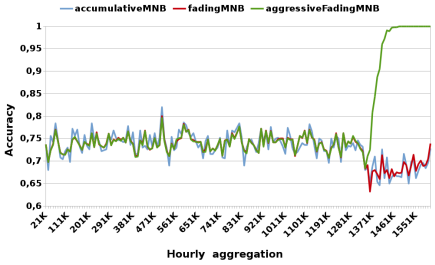

Fig. 5. Prequential evaluation on *accuracy* in the hourly-aggregated stream – same $evalW = 1000$ as in Figure 2, but $\lambda = 0.000003$.

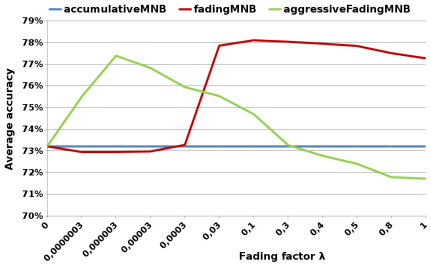

Fig. 6. Effect of fading factor λ on accuracy in the hourly-aggregated stream; prequential evaluation, evaluation window $evalW = 1000$.

the λ increases, its performance drops. This is expected since low values imply that the classifier gradually forgets in a moderate pace, whereas high values mean that the past is forgotten very fast. The performance of *fadingMNB* is the opposite, for very small values of λ there is no actual ageing and therefore the performance is low (and similar to *accumulativeMNB*); whereas as λ increases, *fadingMNB* exploits the ageing of the old data, so its performance improves.

5.5 The Effect of Temporal Granularity and How to Set λ

The temporal granularity of the streams (e.g. hourly, daily, weekly, etc.) and the fading factor λ clearly affect each other. To illustrate this, for a decay degree of $\lambda = 0.1$, the weight of a instance is halved every $\frac{1}{0.1} = 10$ hours, days, weeks for hourly, daily, weekly temporal aggregation of the stream, respectively. The 10 weeks rate might be too low for some fast changing applications, whereas the 10 hours rate might be too high for other applications. Therefore, the choice of the decay factor λ should be done in conjunction with the temporal granularity of the stream. In the subfigures of Figure 7, we show the performance of the ageing-based classifiers for the different temporal-aggregations versions of the stream and different values of λ.

On the left part of Figure 7, we show the accuracy for $\lambda = 0.1$. This value means that an instance *loses half of its weight* after 10 hours, 10 days, 10 weeks for the hourly, daily, weekly aggregated stream respectively. We can see that hourly aggregation deals best with the change in the distributions as it performs best after the distribution changes; closely followed by aggressive fading in a daily aggregation. The worst performance occurs when using weekly aggregation *and* the *fadingMNB*, which indicates that forgetting every 10 weeks is not appropriate in this case; a more frequent forgetting of the data is more appropriate.

We increase the forgetting rate to $\lambda = 0.2$ (mid part of Figure 7), i.e., an instance *loses half its weight* after 5 hours, 5 days, 5 weeks for the hourly, daily, weekly aggregated stream respectively. The results are close to what we observed before; the hourly aggregation has the best performance for this dataset.

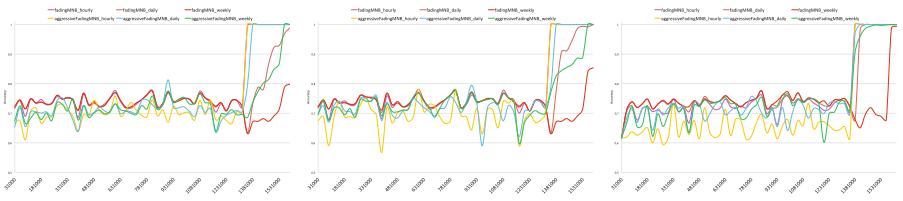

Fig. 7. Prequential accuracy of *fadingMNB*, *aggressiveFadingMNB* in comparison to *accumulativeMNB* (horizontal line) for different levels of granularity in the stream (using $evalW = 1000$): $\lambda = 0.1$ (left), $\lambda = 0.2$ (middle), $\lambda = 1.0$ (right)

The results for a much larger λ, $\lambda = 1.0$, are shown in the right part of Figure 7. A $\lambda = 1.0$ means that an instance *loses half its weight* after 1 hour, 1 day, 1 week for the hourly, daily, weekly aggregated stream respectively. The *aggressiveFadingMNB* performs worse when there is no drift in the stream, because the classifier forgets *too* fast. This fast forgetting though allows the classifier to adapt fast in times of change, i.e., when drift occurs after instance 1,341,000. Among the different streams, in times of stability in the stream, the aggressive fading classifier in the hourly aggregated stream shows the *worst* performance, followed closely by the daily and then the weekly aggregated streams. In times of change however, the behavior is the opposite with the hourly aggregated stream showing the best adaptation rate, because of no memory. Regarding *fadingMNB*, daily and weekly aggregation show best performance in times of stability followed by the hourly aggregated stream. In times of drift *on the other hand*, the hourly aggregation adapts *the fastest*, followed by daily and weekly aggregation streams.

To summarize, the value of λ affects the performance of the classifier over the whole course of the stream. In times of drifts in the stream, a larger λ is preferable as it allows for fast adaptation to the new concepts appearing in the stream. In times of stability though, a smaller λ is preferable as it allows the classifier to exploit already learned concepts in the stream. The selection of λ "works" in collaboration with the stream granularity. Obviously, at a very high granularity (such as a time unit of one hour), lambda can be higher than at a lower granularity (such as a time unit of one week).

6 Conclusions and Outlook

Learning a stream classifier is a challenging task due to the changes in the stream population (at both instance and feature levels) and the necessity for classifier adaptation to change. Adapting to change means that the model should be updated online; this update might mean that existing parts of the model are updated based on new observations, new model parts are added and old model parts are forgotten.

In this work we couple a very popular classifier, Multinomial Naive Bayes, with adaptation-to-change mechanisms. In particular, we introduce the notion of

ageing in MNBs and we derive a gradually fading MNB approach (*fadingMNB*) and an aggressive fading MNB approach (*aggressiveFadingMNB*). We compare our methods to the traditional stream MNB approach (*accumulativeMNB*) and we show its superior performance in an evolving stream of tweets. Our ageing-based approaches recover fast after changes in the stream population, while maintaining a good performance in times of stability, i.e., when no drastic changes are observed. We also show how the fading factor, that regulates the ageing of old data, affects the results and its "connection" to the temporal granularity of the stream.

Our ongoing work involves experimenting with different streams from diverse domains and tuning the fading factor λ online based on the stream, instead of having a constant fading factor over time. As we observed in the current experiments, fast forgetting is important for times of change but in times of stability forgetting should be slower. Another direction of future work encompasses bookkeeping of the model at regular time intervals. In particular, one can maintain a more compact model by removing words that do not contribute much in the classification decision due to ageing, or whose observations in the stream are lower than the expected observations based on their age.

References

1. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: A framework for projected clustering of high dimensional data streams. In: Proceedings of the 30th International Conference on Very Large Data Bases (VLDB), Toronto, Canada (2004)
2. Bermingham, A., Smeaton, A.F.: Classifying sentiment in microblogs: Is brevity an advantage? In: Proceedings of the 19th ACM International Conference on Information and Knowledge Management, CIKM 2010, pp. 1833–1836. ACM, New York (2010)
3. Bifet, A., Frank, E.: Sentiment knowledge discovery in twitter streaming data. In: Pfahringer, B., Holmes, G., Hoffmann, A. (eds.) DS 2010. LNCS, vol. 6332, pp. 1–15. Springer, Heidelberg (2010)
4. Cao, F., Ester, M., Qian, W., Zhou, A.: Density-based clustering over an evolving data stream with noise. In: Proceedings of the 6th SIAM International Conference on Data Mining (SDM), Bethesda, MD (2006)
5. Domingos, P., Pazzani, M.: On the optimality of the simple bayesian classifier under zero-one loss. Mach. Learn. **29**(2–3), 103–130 (1997)
6. Gama, J.A., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Comput. Surv. **46**(4), 44:1–44:37 (2014)
7. Gama, J.: Knowledge Discovery from Data Streams, 1st edn. Chapman & Hall/CRC (2010)
8. Gama, J., Kosina, P.: Recurrent concepts in data streams classification. Knowl. Inf. Syst. **40**(3), 489–507 (2014)
9. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. In: Processing, pp. 1–6 (2009). http://www.stanford.edu/alecmgo/papers/TwitterDistantSupervision09.pdf
10. Gokulakrishnan, B., Priyanthan, P., Ragavan, T., Prasath, N., Perera, A.S.: Opinion mining and sentiment analysis on a twitter data stream. In: Proceedings of 2012 International Conference on Advances in ICT for Emerging Regions (ICTer), ICTer 2012, pp. 182–188. IEEE (2012)

11. Guerra, P.C., Meira, Jr., W., Cardie, C.: Sentiment analysis on evolving social streams: how self-report imbalances can help. In: Proceedings of the 7th ACM International Conference on Web Search and Data Mining, WSDM 2014, pp. 443–452. ACM, New York (2014)
12. Lazarescu, M.: A multi-resolution learning approach to tracking concept drift and recurrent concepts. In: Gamboa, H., Fred, A.L.N. (eds.) PRIS, p. 52. INSTICC Press (2005)
13. Liu, Y., Yu, X., An, A., Huang, X.: Riding the tide of sentiment change: Sentiment analysis with evolving online reviews. World Wide Web **16**(4), 477–496 (2013)
14. McCallum, A., Nigam, K.: A comparison of event models for naive bayes text classification. In: AAAI-98 Workshop on Learning for Text Categorization, pp. 41–48. AAAI Press (1998)
15. Ntoutsi, E., Zimek, A., Palpanas, T., Krger, P., peter Kriegel, H.: Density-based projected clustering over high dimensional data streams. In: Proceedings of the 12th SIAM International Conference on Data Mining (SDM), Anaheim, CA, pp. 987–998 (2012)
16. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Found. Trends Inf. Retr. **2**(1–2), 1–135 (2008)
17. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing, vol. 10, pp. 79–86. EMNLP, ACL, Stroudsburg (2002)
18. Plaza, L., Carrillo de Albornoz, J.: Sentiment Analysis in Business Intelligence: A survey, pp. 231–252. IGI-Global (2011)
19. Sentiment140: Sentiment140 - a Twitter sentiment analysis tool. http://help.sentiment140.com/
20. Sinelnikova, A.: Sentiment analysis in the Twitter stream. Bachelor thesis, LMU, Munich (2012)
21. Sinelnikova, A., Ntoutsi, E., Kriegel, H.P.: Sentiment analysis in the twitter stream. In: 36th Annual Conf. of the German Classification Society (GfKl 2012), Hildesheim, Germany (2012)
22. Turney, P.D.: Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, ACL 2002, pp. 417–424. Association for Computational Linguistics, Stroudsburg (2002)
23. Viera, A.J., Garrett, J.M.: Understanding interobserver agreement: The kappa statistic. Family Medicine **37**(5), 360–363 (2005)
24. Zimmermann, M., Ntoutsi, E., Spiliopoulou, M.: Adaptive semi supervised opinion classifier with forgetting mechanism. In: Proceedings of the 29th Annual ACM Symposium on Applied Computing, SAC 2014, pp. 805–812. ACM, New York (2014)
25. Zimmermann, M., Ntoutsi, E., Spiliopoulou, M.: Discovering and monitoring product features and the opinions on them with OPINSTREAM. Neurocomputing **150**, 318–330 (2015)