Statistical Methods & Applications (2024) 33:235–265
https://doi.org/10.1007/s10260-023-00721-1

ORIGINAL PAPER

Iterative threshold-based Naïve bayes classifier

Maurizio Romano[1] · **Gianpaolo Zammarchi**[1] · **Claudio Conversano**[1]

Accepted: 14 August 2023 / Published online: 5 September 2023

Abstract

The iterative Threshold-based Naïve Bayes (iTb-NB) classifier is introduced as a (simple) improved version of the previously introduced non-iterative Threshold-based Naïve Bayes (Tb-NB) classifier. iTb-NB starts from a Natural Language text-corpus and allows the user to quantify with a numeric value a sentiment (positive or negative) from a specific test. Differently from Tb-NB, iTb-NB is an algorithm aimed at estimating multiple threshold values that concur to refine Tb-NB's decision rules when classifying a text into positive (negative) based on its content. Observations with sentiment scores close to the threshold are marked to be reclassified, hence a new decision rule is defined for them. Such "iterative" process improves the quality of predictions w.r.t. Tb-NB but keeping the possibility to utilize its results as the input of useful post-hoc analyses. The effectiveness of iTb-NB is evaluated analyzing hotel guests' reviews from all hotels located in the Sardinia region and available on *Booking.com*. Furthermore, iTb-NB is compared with Tb-NB in terms of model accuracy, resistance to noise, and computational efficiency.

Keywords Naïve bayes · Post-hoc analysis · Customer satisfaction · Sentiment analysis · Natural language processing · Booking.com

1 Introduction

Nowadays there is an increasing availability of large and complex textual data sets that can be analyzed to extract useful information regarding human behavior. Indeed, the development of methods to perform Natural Language Processing (NLP) as well

✉ Maurizio Romano
romano.maurizio@unica.it

Gianpaolo Zammarchi
gp.zammarchi@unica.it

Claudio Conversano
conversa@unica.it

[1] Department of Economics and Business Sciences, University of Cagliari, Viale Fra Ignazio 17, Cagliari 09123, Italy

as the improvement of their computational capabilities provide an opportunity to make prediction in a variety of fields. To this regard, several platforms that allow to find and book travel services, such as Booking.com, include online reviews that can represent useful instruments for hotels to show their strongest assets and gain consumers' trust. One of the most relevant challenges related to NLP is the identification of the main opinion or sentiment expressed in a text. Sentiment analysis (SA) is widely used to e.g. detect the polarity of a document (Wilson et al. 2005) or identify specific emotions (Yadollahi et al. 2017; Mohammad 2016). The classification of the polarity of product reviews represents a relevant task in SA. Among different approaches that can be used to perform this task, those relying on machine learning methods have become increasingly popular (Fang and Zhan 2015). These methods have been applied to measure SA of texts extracted from social media (Yue et al. 2019; Nguyen et al. 2015; Tavazoee et al. 2020), country reputation (Zammarchi et al. 2023), brand reputation (Vidya et al. 2015) or tourism (Alaei et al. 2019). In tourism, SA is utilized to evaluate opinions of users on specific hotels or destinations, thus allowing to obtain useful information for a business. Indeed, being able to extract complete and relevant data from users' feedback is among the most important aspects to build a marketing strategy (Micu et al. 2017).

Here we focus on the analysis of online reviews to obtain a measure of the satisfaction of clients of hotels. To this aim, we consider reviews from the platform Booking.com. These reviews are composed of two parts which convey information on aspects that the client has identified as positive or negative, respectively. We show how the hereby proposed *Iterative Threshold-based Naïve Bayes classifier* (iTb-NB, Romano et al. (2022a)) introduced as a modification of the original Threshold-based Naïve Bayes classifier (Tb-NB), allows to identify the resulting polarity of a review considered as a whole text. In Romano et al. (2023), it is reported that Tb-NB effectively discriminates positive reviews from negative ones and, at the same time, allows us to quantify the (positive or negative) impact of a specific word within a review. At the same time, the information deriving from Tb-NB can be used to support decision makers as the Tb-NB output can be used further in post-hoc analyses to evaluate different facets of customer satisfaction (Romano et al. 2022b). Last but not least, it has been showed the Tb-NB is preferable to other methods used in SA in terms of classification accuracy, resistance to noise and computational efficiency. In this paper, we follow a similar line of research to show that iTb-NB is an relevant improvement of Tb-NB. In fact, considering that Tb-NB classifies reviews as positive or negative according to a specific threshold deriving from a set of data-driven defined scores, it could become unstable for observations whose score is located in the neighborhood of the estimated threshold. Hence, iTb-NB reduces the possible instability of the Tb-NB classification rule for observations whose score is located in this neighborhood. As will be better described in Sect. 3, this will be done by finding iteratively multiple thresholds and decision rules, that are all defined by refining the interval containing a proportion of (presumed) unstable cases.

The remaining of the paper is as follows. Before introducing formally iTb-NB (Sect. 3), we recall Tb-NB in Sect. 2. Next, we describe the Booking.com reviews data (Sect. 4) that is analyzed to compare iTb-NB with Tb-NB and with alternative methods in Sect. 5, evaluating accuracy (Sect. 5.1), resistance to noise (Sect. 5.2), and computational efficiency (Sect. 5.3) of each classifier. Section 5.4 shows benchmarking results, and Sect. 6 ends the paper with some concluding remarks.

2 Threshold-based naïve Bayes classifier

A Bayesian classifier assigns an observation to the most likely class, based on the values of a set of independent variables. The Naïve Bayes (NB) is a conditional probability model based on the Bayes' theorem. NB assumes independence among predictors, and is able to achieve competitive categorization accuracy, robustness and computing efficiency (Webb et al. 2010). Several important issues concerning NB, like for instance the discretization of input variables (Zhang et al. 2023), the structure of classifier boundaries (Karr et al. 2022), and the weighting of observations (Zhang et al. 2021) have been considered in literature.

One of the main tasks when dealing with textual data is to estimate the probability for each text to belong to one of the response classes: in sentiment analysis we are mainly interested in assessing whether a text is positive or negative. *Threshold-based Naïve Bayes* (Tb-NB, Romano et al. (2023)) is a novel data-driven methodology based on the NB classifier, but without the need to specify the distribution of words or response classes. In fact, NB works by estimating probabilities using words frequency (assuming word independence) and classes representation. In practice it is assumed that, when the amount of available data is large enough, the observed classes representation tends asymptotically to the true classes' representation (which remains unknown). The power of Tb-NB lies in the ability of setting the decision rule using a threshold directly estimated from the data. As a result, once the training phase is completed, each new text is assigned to the most likely class.

Tb-NB is essentially a binary classifier, since in many real-world applications people are asked to express a binary choice: positive or negative, pro or con, and so on. To train the Tb-NB classifier, the total number of text-corpus $n_{\mathcal{R}}$ is split into a training set of size n_r and a test set of size $n_{\mathcal{R}} - n_r$, where $\mathcal{R}$ is the corpora created using all the available text-corpus.

$$\mathcal{R} = \{r_1, \dots, r_j, \dots, r_n\}, \qquad j = 1, \dots, n_r, (n_r + 1), \dots, n_{\mathcal{R}}.$$

After a data cleaning step (see Sect. 4), each of the n_w words in the $n_{\mathcal{R}}$ text-corpus are used to create a Bag-of-Words (BoW) such as

$$\mathcal{W} = \{w_1, \dots, w_i, \dots w_{n_w}\}, \qquad i = 1, \dots, n_w.$$

Tb-Nb is a supervised classifier, thus it requires the presence of a labeled response variable at least in the training set observations. Alternatively, it results particularly suitable when each text-corpus is composed of two sub-corpus: a positive sub-corpus c_j^+ and a negative sub-corpus c_j^-. This is the case, among others, of the Booking.com data where each single review is composed of at least one among a positive comment and a negative comment about the specific accommodation/service. Since not all text-corpus have both sub-corpus, the occurrence of the empty set ∅ has to be considered when joining the words of the positive sub-corpus c_j^+ to the words in the negative sub-corpus c_j^- of the specific text-corpus. Therefore, the content of a generic text-corpus r_j is

$$r_j = (c_j^+ \cup c_j^-) = \left\{w_1, \dots, w_k, \dots w_K\right\} \qquad \text{with } (w_1, \dots, w_k, \dots w_K) \in \mathcal{W}$$

Let $\Lambda(\cdot)$ be a scoring function computed for each word w_k of the n_r text-corpus used to train the Tb-NB classifier. This scoring function exploits the Bayes' theorem and is computed using a probability function $\pi(\cdot)$ as follows

$$\begin{aligned}\Lambda\Big(w_k|(c_j^+, c_j^-) \in r_j\Big) = & \log\left[\frac{\pi(c_j^+|w_k)}{\pi(c_j^-|w_k)}\right] = \\ & = \log\left[\frac{\pi(w_k|c_j^+)}{\pi(w_k|c_j^-)} \cdot \frac{\pi(\bar{w}_k|c_j^+)}{\pi(\bar{w}_k|c_j^-)} \cdot \frac{\pi(c_j^+)}{\pi(c_j^-)}\right] = \\ & = \underbrace{\Big[\log \pi(w_k|c_j^+) - \log \pi(w_k|c_j^-)\Big]}_{\mathcal{L}(w_k)} \\ & + \underbrace{\Big[\log \pi(\bar{w}_k|c_j^+) - \log \pi(\bar{w}_k|c_j^-)\Big]}_{\mathcal{L}(\bar{w}_k)} \\ & + \Big[\log \pi(c_j^+) - \log \pi(c_j^-)\Big] \approx \\ & \approx \mathcal{L}(w_k) + \mathcal{L}(\bar{w}_k)\end{aligned} \tag{1}$$

The scoring function $\Lambda(w_k|c_j^+, c_j^-)$ is computed for each word w_k included in the j-th text-corpus r_j. It provides a numerical value that allows to understand if it is more likely that w_k appears in the positive sub-corpus c_j^+ or vice-versa. Equation 1 highlights that $\Lambda(w_k)$ is composed of two parts. The first is a function $\mathcal{L}(w_k)$ that computes the the log-likelihood ratio of the event $(w_k \in r_j)$ and assesses how likely a word w_k is present in a text-corpus. The second part is a function $\mathcal{L}(\bar{w}_k)$ that assesses how likely w_k is not present in the same text by computing the log-likelihood ratio of the event $(w_k \notin r_j)$. The term $\Big[\log \pi(c_j^+) - \log \pi(c_j^-)\Big]$ in Eq. 1 corresponds to the proportions of observed positive (or negative) sub-corpus in the corpora $\mathcal{R}$ and is not considered as it is constant for all the w_k in $\mathcal{W}$.

The log-likelihood scores can be computed for the whole text-corpus (including both the positive and negative components) as well as for single sub-corpus (i.e. only positive or negative sub-corpus). In the first case, by computing the scoring function $\Lambda(w_k)$ for all the K words included in a text-corpus r_j $(j = 1, \dots, n_r)$, it is possible to assess the polarity of the text-corpus r_j as

$$\begin{aligned}\Lambda(r_j) &= \Lambda(w_1, \dots, w_k, \dots w_K) = \sum_{k=1}^{K} \Lambda\Big(w_k | (c_j^+, c_j^-) \in r_j\Big) \\ &= \sum_{k=1}^{K} \mathcal{L}(w_k) + \mathcal{L}(\bar{w}_k)\end{aligned} \tag{2}$$

In this case, the Tb-NB classifier is trained on the n_r text-corpus composing the training set and the class label (positive or negative polarity) is predicted for the $n_{\mathcal{R}} - n_r$ text-corpus composing the test set.

In the second case, it is possible to quantify if a sub-corpus $c_j = c_j^+ \cup c_j^-$ $(j = 1, \dots, n_c)$ has a negative or a positive polarity by computing the scoring function $\Lambda(w_m)$ for all the M words included in it

$$\begin{aligned}\Lambda(c_j) &= \Lambda(w_1, \dots, w_m, \dots w_M) = \sum_{m=1}^{M} \Lambda\Big(w_m | (c_j^+, c_j^-) \in c_j\Big) \\ &= \sum_{m=1}^{M} \mathcal{L}(w_m) + \mathcal{L}(\bar{w}_m)\end{aligned} \tag{3}$$

with $(w_1, \dots, w_m, \dots w_M) \in c_j \in \mathcal{W}$. In this case, the n_c sub-corpus included in the n_r text-corpus composing the training set are used to train the Tb-NB classifier, whilst the remaining sub-corpus included in the $n_{\mathcal{R}} - n_r$ text-corpus composing the test set are used to assess its predictive accuracy.

Either for the task of predicting the polarity of a whole text-corpus or predicting that of a specific sub-corpus, the log-odds ratio $\Lambda(w_k)$ (Eq. 1) is computed for all the words $w_k \in \mathcal{W}$. Next, the values of $\Lambda(w_k)$ corresponding to the words included in each text-corpus r_j $(j = 1, \dots, n_r)$ based on Eq. 2, or the words included in each comment c_j $(j = 1, \dots, n_c)$ based on Eq. 3, are summed to obtain $\Lambda(r_j)$ or $\Lambda(c_j)$, respectively.

In both cases, the predicted class assignment is based on a decision rule $\mathcal{D}$ that assigns a positive or negative polarity to the text-corpus r_j^* or to the sub-corpus c_j^* included in the test set. The decision rule $\mathcal{D}$ is specified according to a threshold parameter τ estimated from data. This threshold parameter corresponds to a specific value of the log-odds ratio $\Lambda(\cdot)$ that we use to classify a text-corpus r_j^*, or a sub-corpus c_j^*, as positive or negative. Formally, the decision rule $\mathcal{D}$ for the text-corpus r_j^* is defined as

$$\mathcal{D}_{r_j^*} : \begin{cases} \Lambda\Big(r_j^*\Big) > \hat{\tau} \ \rightarrow r_j^* = +1 \\ \Lambda\Big(r_j^*\Big) \le \hat{\tau} \ \rightarrow r_j^* = -1 \end{cases} \qquad (j^* = n_r + 1, \dots, n_{\mathcal{R}}) \tag{4}$$

In the case of a sub-corpus, the decision rule is defined in a similar way by replacing r_j^* with c_j^*, in Eq. 4. The threshold τ represents the only parameter of the Tb-NB classifier and is estimated based on the training data. In a binary classification setting,

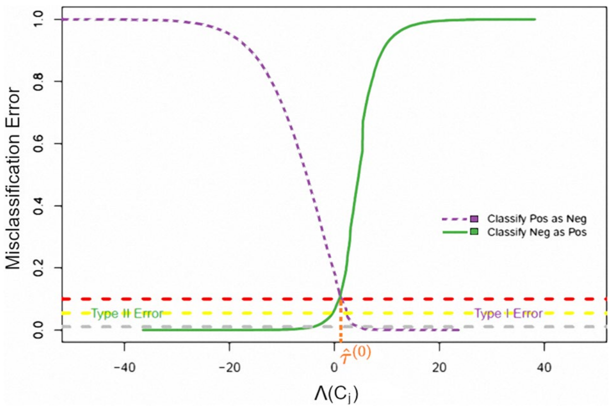


Fig. 1 Decision rule of Tb-NB (Eq. 4) based on the minimization of both Type I and Type II errors obtained from k-fold cross validation on training data. The estimated τ is found as the value of $\Lambda(c_j)$ minimizing both Type I and Type II errors

we can estimate a threshold τ to minimize either the Type I error, the Type II error or both, according to the specific task and aims of the analysis. In any case, the threshold τ is estimated by applying k-fold cross-validation on the n_r text-corpus, or to the corresponding number of sub-corpus, part of the training set. As an example, Fig. 1 shows the distribution of the Type I and Type II errors obtained by applying k-fold cross validation on the training data.

To estimate Type I and Type II errors, "positive" or "negative" labels of the sub-corpus included in the training data are available, while the same is not true in case the reference text is a whole text-corpus. In such a case, an unsupervised learning task can be turned into a supervised learning task by including a priori information on the polarity of a text using external sources or other methods such as, for example, context-based word embeddings (Yu et al. 2018).

3 Iterative threshold-based Naïve bayes classifier

Due to its simplicity and ease of implementation, NB is among the most used classifiers and its performance is often compared with that of other classifier including more advanced methods. In particular, NB is one of the most used classifier in sentiment analysis (see, for example, Denecke and Deng (2015), Dey et al. (2016)), and in experiments concerning the measurement of customer satisfaction based on reviews (Sánchez-Franco et al. 2019; Laksono et al. 2019; Xu et al. 2020; Tran et al. 2019).

As demonstrated in Romano et al. (2023), the Tb-NB classifier overperforms NB and other classifiers in measuring customer satisfaction of hotels' guests whose reviews are reported on the Booking.com website. Anyway, Tb-NB might become unstable if the decision boundary derived with the estimated threshold does not clearly separate text-corpus that have to be classified as "positive" from those that have to be classified as "negative". The set of scores $\Lambda\left(r_j\right)$ (Eq. 2), or $\Lambda\left(c_j\right)$ (Eq. 3) are compared with the estimated threshold $\hat{\tau}$ to decide if a text-corpus r_j (or a sub-corpus c_j) has to be classified as positive or negative (Eq. 4). Of course, the decision rule is likely to be more reliable as long as the computed value of $\Lambda(\cdot)$ is large compared to $\hat{\tau}$. Contrariwise, if the computed value of $\Lambda(\cdot)$ is close to $\hat{\tau}$ the decision rule $\mathcal{D}$ is less reliable.

The "augmented" *iterative Threshold-based Naïve Bayes* (iTb-NB) classifier is hereby introduced to reduce the possible instability of the Tb-NB classification rule for estimated value of $\Lambda(\cdot)$ located in the neighborhood of $\hat{\tau}$. iTb-NB introduces additional steps in the estimation of the sentiment polarity. It iteratively considers a subset of observations whose estimated $\Lambda(\cdot)$ is in the neighborhood of $\hat{\tau}$ and estimates a new τ for this subset of cases only. Thus, the value of $\hat{\tau}$ for observations located near the original $\hat{\tau}$ is refined iteratively in order to reduce uncertainty characterizing these observations.

Operationally, for $\Lambda\left(r_j\right)$, or $\Lambda\left(c_j\right)$ scores located in a neighborhood of $\hat{\tau}$, say $\omega(\hat{\tau})$, an additional step that refines the threshold while classifying only those r_j or c_j that are close to $\hat{\tau}$ is introduced. Then, we classify again those r_j or c_j with a new estimated value of τ. The "augmented" iterative Threshold-based Naïve Bayes works as follows:

1. a proportion ω of observations close to the $\hat{\tau}$ originating from the Tb-NB classifier, and located in the uncertainty area, are marked for being reclassified;
2. A new decision rule is created for those observations only, estimating a new value of τ, denoted as $\hat{\tau}^{(i)}$;
3. Observations located in a neighbourhood of $\hat{\tau}^{(i)}$, i.e.: in the updated uncertainty area, are remarked for being reclassified and Steps 1–2 are repeated until the proportion of cases in the uncertainty area reduces to zero.

Focusing on the scores $\Lambda\left(c_j\right)$, but the same approach can be used for the scores $\Lambda\left(r_j\right)$, the iTb-NB algorithm is formalized in Algorithm 1.

iTB-NB requires as input parameters the value of $\hat{\tau}$ and the decision rule D_{c_j}, both obtained from Tb-NB, together with the set of comments ω to be reclassified. These are the comments included in the interval $\hat{\tau} \pm \delta\omega$. The iterations of iTB-NB proceeds as long as the number of comments included in ω is at least equal to the minimum number of reclassifying comments s, which is another user-defined input parameter.

If the number of comments included in ω is lower than the minimum number of reclassifying comments s, there is no iteration and the decision rule is that of Tb-NB (one threshold, one decision rule). Thus, Tb-NB results in a special case of iTB-NB.

Instead, if the number of comments in ω is at least equal to the s, the iterative assignment process of iTB-NB takes place. In each iteration i, iTB-NB separates positive comments belonging to $\omega^{(i)}$ from negative ones, and computes for the two sets of comments the empirical distributions $f^{+}(\cdot)$ and $f^{-}(\cdot)$ of $\Lambda(c_j)$. As reported in point 4 of Algorithm 1, the position of the two distributions $f^{+}(x)$ and $f^{-}(x)$ allows to determine a proper $\hat{\tau}^{(i)}$ and $D_{c_j}^{(i)}$ for the set of comments in $\omega^{(i)}$.

```
Algorithm 1 iterative Threshold-based Naive Bayes (iTb-NB) algorithm
Input:
$i = 0$;    $s = minsize$;    $\hat{\tau}^{(0)} = \hat{\tau}$;
$\omega^{(0)} = \{c_j \in \hat{\tau}^{(0)} \pm \delta\omega\}$;    $D^{(0)}_{c_j} = D^{*}_{c_j}$;    $j = (1, \dots, n_c)$;

  while $|\omega^{(i)}| \geq s$ do
    1. $i = i + 1$
    2. $\varphi_j = \begin{cases} 0 & \text{if } c_j = c_j^- \\ 1 & \text{if } c_j = c_j^+ \end{cases}$;    $x = \{\Lambda(c_j) : c_j \in \omega^{(i)}\}$;    $f(x) = \frac{d}{dx} F_x(x)$;
    3. Map $f(x)$ into $f^+(x)$ and $f^-(x)$:
       $\forall x \in (-\infty; +\infty) \quad f^+(x) = \{x : \varphi_j = 1\} \quad f^-(x) = \{x : \varphi_j = 0\}$
    4. Compute:
       $x^+_{max} = \underset{x \in (-\infty;+\infty)}{\text{argmax}} \; f^+(x) \qquad x^-_{max} = \underset{x \in (-\infty;+\infty)}{\text{argmax}} \; f^-(x)$
       $\hat{\tau}^{(i)} = \begin{cases} \underset{x \in (x^-_{max}; x^+_{max})}{\text{argmin}} \; (f^+(x) - f^-(x)) & \text{if } x^+_{max} > x^-_{max} \\ \underset{x \in (x^+_{max}; x^-_{max})}{\text{argmin}} \; (f^+(x) - f^-(x)) & \text{if } x^+_{max} < x^-_{max} \\ \hat{\tau}^{(i-1)} & \text{if } x^+_{max} = x^-_{max} \end{cases}$
       $D^{(i)}_{c_j} : \begin{cases} x \leq \hat{\tau}^{(i)} \;\&\; x^+_{max} > x^-_{max} & \rightarrow c^*_j = -1 \\ x > \hat{\tau}^{(i)} \;\&\; x^+_{max} < x^-_{max} & \rightarrow c^*_j = -1 \\ x > \hat{\tau}^{(i)} \;\&\; x^+_{max} > x^-_{max} & \rightarrow c^*_j = +1 \\ x \leq \hat{\tau}^{(i)} \;\&\; x^+_{max} < x^-_{max} & \rightarrow c^*_j = +1 \end{cases}$
    5. Set $\omega^{(i)} = \{c_j \in \hat{\tau}^{(i)} \pm \delta\omega\}$
Output: $\{\omega^{(0)}, \dots, \omega^{(i)}\}$;    $\{\hat{\tau}^{(0)}, \dots, \hat{\tau}^{(i)}\}$;    $\{D^{(0)}_{c_j}, \dots, D^{(i)}_{c_j}\}$.
```

Algorithm 1 iterative Threshold-based Naive Bayes (iTb-NB) algorithm

The specific features of these iterations are better explained with an example in Fig. 2. The maximum values x^+_{max} and x^-_{max} of $f^+(x)$ and $f^-(x)$ are computed to understand in which tail of the distribution of $f^+(x)$ the estimating $\tau^{(i)}$ is located. In Fig. 2 (top panel) it is assumed that $x^+_{max} > x^-_{max}$, but the reverse relationship is also possible, thus the new value of τ to be estimated ($\hat{\tau}^{(1)}$) is located on the left tail of $f^+(x)$ in the uncertainty area $\omega^{(1)}$ (highlighted in yellow) and is equidistant from x^+_{max} and x^-_{max}. A proper decision rule $D^{(1)}_{c_j}$ is specified for cases in $\omega^{(1)}$ only, and this iterating step is repeated until the two distributions $f^+(x)$ and $f^-(x)$ are completely (or almost completely) overlapped (Fig. 2, bottom panel). Hence, at the end of the

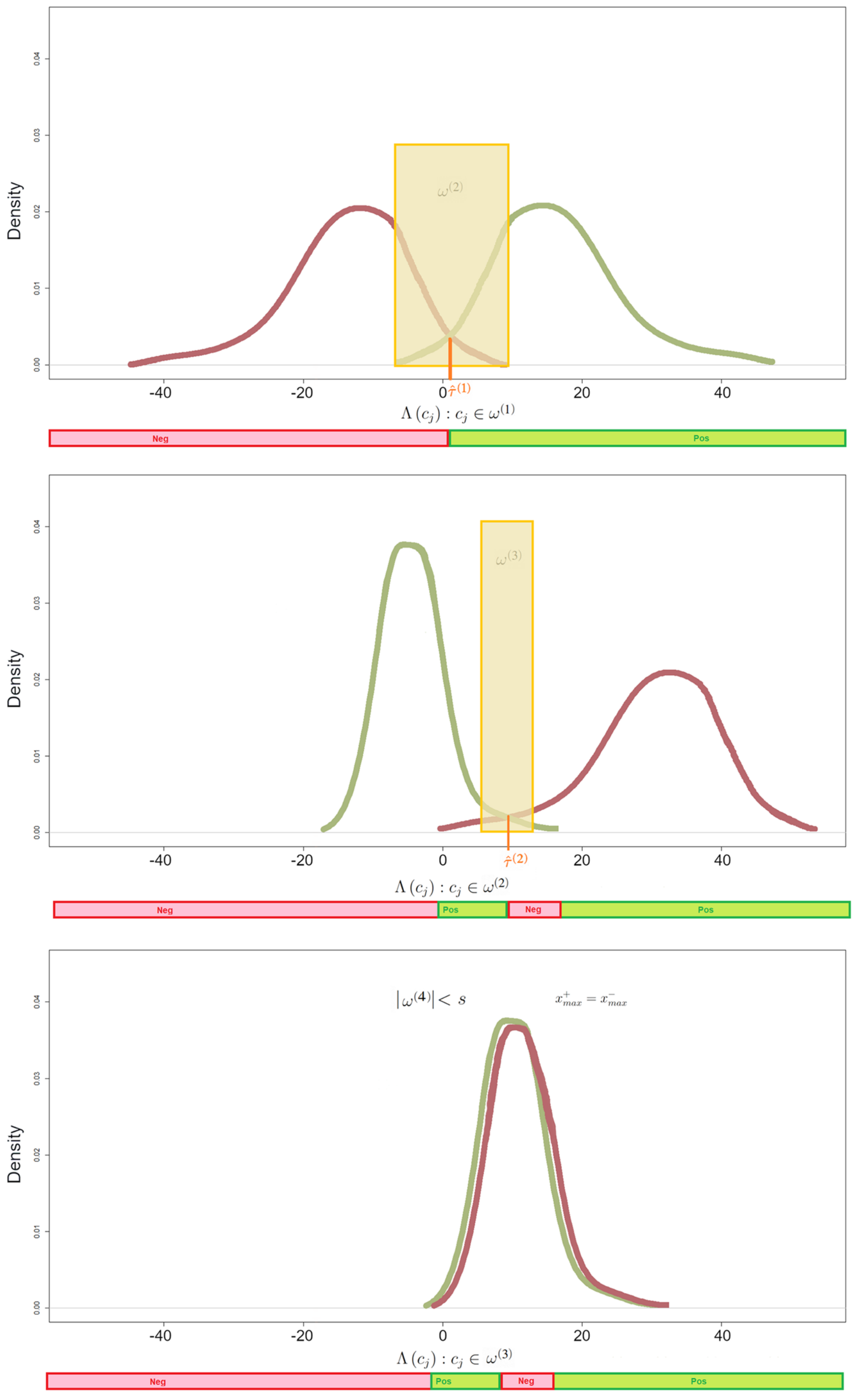


Fig. 2 An example of the features of the iTb-NB iterations (points 1 to 5 of Algorithm 1)

iterations we have two thresholds with an associated decision rule for each one. Thus, the classification of c_j is done as follows:

$$c_j = \begin{cases} +1 \text{ if } \Lambda\left(c_j\right) \in \omega^{(2)} \leq \hat{\tau}^{(2)} \text{ or } \Lambda\left(c_j\right) \in \omega^{(1)} > \hat{\tau}^{(1)} \\ -1 \text{ if } \Lambda\left(c_j\right) \in \omega^{(2)} > \hat{\tau}^{(2)} \text{ or } \Lambda\left(c_j\right) \in \omega^{(1)} \leq \hat{\tau}^{(1)} \end{cases}$$

In other words, in that case, the most central $\Lambda(\cdot)$ values will follow the decision rule $\mathcal{D}^{(2)}$ (that uses $\hat{\tau}^{(2)}$) whilst the more extreme values will follow $\mathcal{D}^{(1)}$ (that uses $\hat{\tau}^{(1)}$).

The iTB-NB algorithm returns the sequences of estimated τ and decision rules D_{c_j} for each set of comments $\omega^{(i)}$ identified in each iteration i. As a result, the proportion of observations marked for being re-classified is estimated while considering the uncertainty area around τ. Using a large value of ω leads to results that are similar to those obtained from Tb-NB classifier, whilst using a low value for ω might lead to consider few observations only, thus producing a decision rule which overfits the data.

Whereas, the stopping criterion of the iterative Tb-NB depends on at least one of the following conditions:

1. the number of cases (positive and negative comments) to be reclassified in a specific iterations is lower than the user-defined minimum size s;
2. there is no intersection point (other than zero) between $f^{+}(x)$, and $f^{-}(x)$ and $x^{+}_{max} \neq x^{-}_{max}$. The two distributions are well separated;
3. $x^{+}_{max} = x^{-}_{max}$. The two distributions $f^{+}(x)$ and $f^{-}(x)$ are not distinguishable, thus there is no geometric solution to the mapping of the two functions.

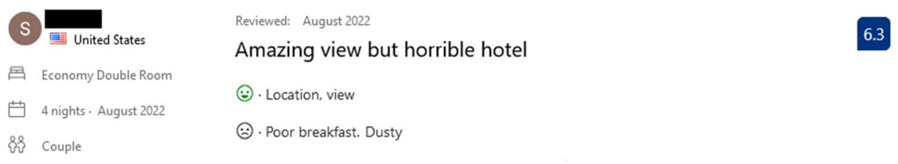
In all cases, the τ estimated by the Tb-NB classifier at iteration zero, or the set of $\hat{\tau}$s obtained in the previous iterations of iTB-NB, are the output of iTB-NB.

Moreover, it is interesting to highlight that empirical results on the Booking.com data (Sect. 4) suggest that the stopping criterion usually is reached after no more than two iterations ($0 \leq i \leq 2$), a suitable value for the minimum size is $s = 15$ whilst specifying ω so that it includes 20% of previously classified comments consistently improves the accuracy of the standard Tb-NB classifier (Sect. 5).

4 Booking.com reviews dataset

We consider the same dataset utilized in Romano et al. (2023) to allow for comparability of results obtained from iTb-NB with those of previous analyses. Moreover, this dataset is chosen because the reviews available on Booking.com come from customers who effectively stayed in a hotel and thus from consumers who have actually used the specific (accommodation) service.

The original reviews are scraped from the Booking.com website and collected in a specific dataset to which a data cleaning process is applied before training the iTb-NB classifier and check if it is able to improve the sentiment prediction capability of Tb-NB. We collect data about the reviews containing opinions about all the 619 Sardinian hotels offering accommodations on Booking.com. Each review includes



S [redacted]
United States

Economy Double Room

4 nights · August 2022

Couple

Reviewed: August 2022

Amazing view but horrible hotel

6.3

☺ · Location, view

☹ · Poor breakfast. Dusty

Fig. 3 Example of a review on Booking.com. The first comment (☺) is positive whilst the second one (☹) is negative. A reviewer might decide to leave just one of them. A review is the union of the two comments provided by a reviewer

two parts regarding the client's experience with the hotel, one on positive and one on negative aspects. Figure 3 shows an example of a review recently posted on the Booking.com website. Considering that "text-corpus" is intended as any amount of Natural Language text, in this context the "text-corpus" are the reviews, whilst the "sub-corpus" are the comments.

Reviews have been collected using a Python extractor that relies on three libraries: Requests allows to send HTTP/1.1 requests;[1] BeautifulSoup allows web scraping;[2] and Parallel allows the two other libraries to work together.[3] The extractor allows to collect all data available on the platform after specifying a destination and the reference dates. A total of 66,237 reviews, posted from January 3 2015 to May 27 2018, for the 619 hotels located in Sardinia was collected. The dataset includes 106,800 comments (62,291 positive and 44,509 negative) in Italian (86.14%) or English (13.86%).

The raw data has been cleaned with respect to: (1) emoji and emoticon textual replacement; (2) removal of links, acronyms, keywords or alphanumeric characters, stop words, and punctuation. Conversely, consistent with Chai (2019) and Morante and Blanco (2021), negative words like "not" have not been removed as their elimination can alter the meaning as well as the sentiment of a text. The output of the data cleaning process is the Bag-of-Words representation of the reviews, and the related comments, posted on Booking.com

We apply the iTb-NB classifier to the cleaned Booking.com data to evaluate its accuracy. Following the basic steps of Tb-NB and iTb-NB, described in Sect. 2 and Sect. 3, we compute the log odds ratio (Eq. 1) for each word w_k included in the Bag-of-Words $\mathcal{W}$ as well as for each comment c_j (Eq. 3). An example of the values of some of the components of the score function $\Lambda(c_j)$ specified in Eq. 3 is shown in Table 1, which reports the value of both the probabilities and the scoring functions computed for the reviews, and the two related comments, shown in Fig. 3.

To recall how the polarity classification of comments and reviews works, we hereby show how to classify both comments and the entire review represented in Fig. 3. For this purpose, we define a specific review r such that $r = \{c^+ \cup c^-\}$, with $c^+ = \{location, view\}$, and $c^- = \{poor, breakfast, dusty\}$. Based on the values

[1] https://pypi.org/project/requests/.

[2] https://pypi.org/project/beautifulsoup4/.

[3] https://joblib.readthedocs.io.

Table 1 Threshold-based Naïve Bayes output

	Location	View	Poor	Breakfast	Dusty	..
$\pi(w_k\|c_j^-)$	0.016	0.019	0.023	0.177	0.010	..
$\pi(w_k\|c_j^+)$	0.114	0.087	0.001	0.274	0.001	..
$\mathcal{L}(w_k)$	1.937	1.544	−4.199	0.437	−3.421	..
$\mathcal{L}(\bar{w}_k)$	−0.105	−0.072	0.023	−0.126	0.010	..

reported in Table 1, we apply Eq. 2 and Eq. 3 to compute the value of the scoring function as follows:

$$\begin{aligned}\Lambda(c^+) =& \mathcal{L}(location) + \mathcal{L}(view) + \mathcal{L}(\overline{poor}) + \mathcal{L}(\overline{breakfast})\\ &+ \mathcal{L}(\overline{dusty}) + \cdots = 0.141\\ \Lambda(c^-) =& \mathcal{L}(poor) + \mathcal{L}(breakfast) + \mathcal{L}(dusty) + \mathcal{L}(\overline{location})\\ &+ \mathcal{L}(\overline{view}) + \cdots = -4.469\\ \Lambda(r) =& \mathcal{L}(location) + \mathcal{L}(view) + \mathcal{L}(poor) + \mathcal{L}(breakfast)\\ &+ \mathcal{L}(dusty) + \cdots = -1.144\end{aligned}$$

Moreover, since for the observed data the estimated τ is $\hat{\tau} = 1.138$ (see Sect. 5.4 for details), the text corpora r and the two sub-corpus c^- and c^+ are classified into "positive" (+1) or "negative" (−1) according to Eq. 4:

$$\begin{aligned}\Lambda(c^+) =& 0.141 > \hat{\tau} \rightarrow +1\\ \Lambda(c^-) =& -4.469 \leq \hat{\tau} \rightarrow -1\\ \Lambda(r) =& -1.144 \leq \hat{\tau} \rightarrow -1\end{aligned}$$

Both comments are correctly classified by Tb-NB, whilst we see the polarity assigned to the entire review is "negative".

5 Benchmarking iTb-NB

The performance of iTb-NB is compared with Tb-NB and with that of other well-known classifiers, in particular with the same classifiers considered in Romano et al. (2023) to ensure that results are directly comparable with the previous ones: Logistic Regression (LOG), Random Forest (RF), standard Naïve Bayes (NB E1071), Naïve Bayes using kernel estimated densities (NB KlaR), Decision Trees (CART), Linear Discriminant Analysis (LDA), and Support Vector Machine (SVM). Moreover, we included Neural Networks (NN) among the set of on-the-shelf classifiers. Specifically, we use the R package "neuralnet" (Fritsch et al. 2019) which implements resilient backpropagation (RPROP) with weight backtracking (Riedmiller and Braun 1993) as the default algorithm. Following the comparative analysis done by Dogan and Tanrikulu (2013), comparisons are based on three main factors: ability to correctly classify a positive (negative) comment (Sect. 5.1), robustness (Sect. 5.2), and computing time required for both fitting the model and predicting the text polarity (Sect. 5.3). Those aspects are analyzed more in detail in what follows.

5.1 Accuracy

We apply 5-fold cross-validation to estimate the threshold parameter τ on the entire set of 106,800 comments through the decision rule specified in Eq. 4. We estimate $\Lambda(c_j)$ for the observations included in the original data but not in the considered k-th fold ($k = 1, \ldots, 5$) and compute the Misclassification Error (ME) for observations included in the k-th fold. The Tb-NB's estimated τ is that minimizing simultaneously both the Type I and the Type II errors, as both errors (classifying a comment as positive when it is negative, or vice versa) are considered as equally important in this particular type of analysis.

We compare the performance of iTb-NB with Tb-NB and with that of competitors using different classification performance metrics: accuracy, sensitivity (True Positive Rate), specificity (True Negative Rate), F1 score, and Matthews' correlation coefficient (Chicco and Jurman 2020).

To further enforce the validity of the results obtained on the whole dataset we consider how classifiers accuracy varies when changing the dataset size as well as the size of both training and test sets. To this purpose, we re-estimate the classifiers repeating the analysis several times based on the following factors:

a) the total sample size n: 20,000; 50,000 and 100,000;
b) the training-test set proportions: 50–50, 67–33, 80–20;
c) the selection of comments with more than three words only.

The last factor is considered because preliminary trials seem to indicate that removing short comments (up to three words) improves the classification accuracy of both Tb-NB and iTb-NB. For each combination of total sample size $\times$ training-test set proportions $\times$ elimination of short comments we estimate the different classifiers 100 times on resampled versions of the original data and compute the performance metrics.

5.2 Noise resistance

In many real-world sentiment analysis applications, the goal is to classify text as *positive* ($y = +1$) or *negative* ($y = -1$). In general, it would be easier for any classifier used in the analysis to distinguish between positive and negative text if the two conditional distributions $\mathbf{X}|(y = -1)$ and $\mathbf{X}|(y = +1)$ are well separated in nature. To quatify noice resistance, we focus on the performance of iTb-NB, and of the other classifiers used as benchmark, when data are altered on purpose. Under this paradigm, we measure the noise resistance of comparable classifiers as their capacity to be as accurate as possible when the two conditional distributions indicated above overlap to some extent. In our experiment, overlapping arises as the original class labels are swapped randomly. We utilize once more the same data retrieved from

Booking.com, but changing the label of a certain percentage of comments from positive to negative, and vice-versa, keeping the original text unchanged. We proceed by creating different versions of the corpus with an increasing proportion of perturbed data. Specifically, the amount of perturbed data is set to 25%, 33% and 50%, respectively.

5.3 Computational efficiency

Computational efficiency of a classifier is another important aspect that must be taken into consideration when a very large number of text-corpus or sub-corpus needs to be classified. Under some circumstances, e.g. when results obtained in real-time are of primary importance for decision makers, it is worth considering if a small amount of prediction accuracy might be sacrificed in place of an increase in computing time. In this respect, we evaluate if iTb-NB is able to contemplate computing time and accuracy when classifying text-corpus as positive or negative. To this purpose, we randomly extract 100 text-corpus from the original Booking.com data and divide the sample into a learning set and a test set of equal size. Next, we measure the computing time required to train the model on the learning set and that required to predict new cases included in the test set. Computational efficiency is measured by repeating 100 times the above-described experiment and is evaluated for all the classifiers involved in the benchmarking experiments.

5.4 Results of the benchmarking experiments

All the classifiers mentioned in Sect. 5 were used to predict the polarity of comments included in the Booking.com dataset, and the predictive accuracy of each classifier was measured using 5-fold cross-validation.

Considering Tb-NB, the selection the estimated τ as the one minimizing both Type I and Type II errors leads to a value of $\hat{\tau} = 1.138$ through which the Tb-NB classifier is able to classify correctly 91.1% of the out-of-fold instances. Whereas, applying the refining process of the classification rule of Tb-NB to the 20% of observations close to $\hat{\tau}$ (Algorithm 1), the percentage of correctly classified out-of-fold instances increases to 92.5% with iTB-NB.

Furthermore, we computed all the performance metrics listed in Sect. 5.1 for all the considered classifiers. Results are visualized in Fig. 4, whilst numerical values of the performance metrics are reported in Table 4 (Appendix).

In Fig. 4, all the performance metrics have been rescaled in [0, 1] to both facilitate visual comparisons and compute an average score which is also reported in the plot. Figure 4 clearly shows that iTb-NB is the most accurate classifier with respect to four out of the five performance metrics. Notably, iTB-NB provides a Matthews correlation coefficient (Accuracy) of 0.829 (0.925) versus an average value of 0.469 (0.826) obtained from the alternatives.

As for changing the training-test set proportions, for the sake of brevity in this section we report results concerning the 80–20 case only (Table 2).They indicate that, although iTb-NB is never the best nor the worst performing classifier, it

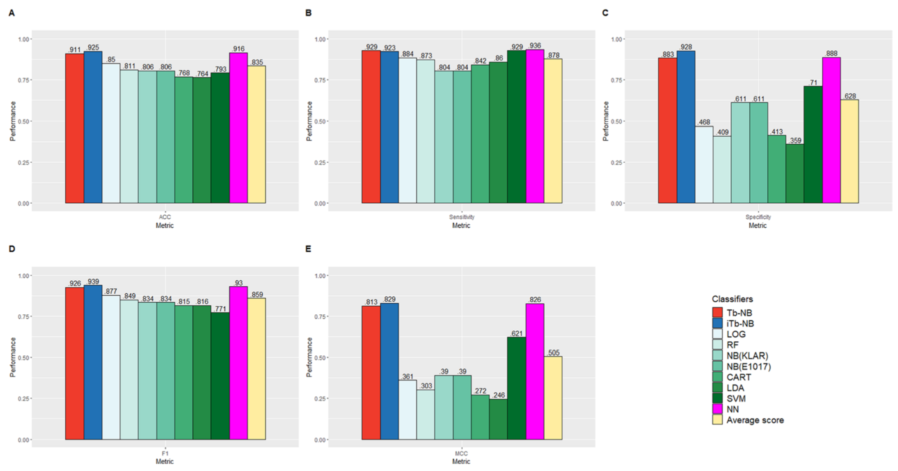


Fig. 4 Performance metrics. Source: Table 4

provides values of classification accuracy metrics that are in line with those obtained by other classifiers. This finding further enforces the strength of Tb-NB and iTb-NB as they provide results that are easily interpretable and usable.

Results obtained for the other training-test set proportions are very similar to those of Table 2 and are reported in the Appendix (Tables 2bis (Table 5) and 2ter (Table 6)).

Results of the noise resistance tests (Sect. 5.2) are visualized in Figs. 6, 7 and 5, whilst numerical values of the performance metrics are reported in Table 7 (Appendix).

Figure 5 shows the values of the classification performance metrics in the case the percentages of perturbed data are equal to 0% (Panel A), 25% (Panel B), 33% (Panel C), and 50% (Panel D), respectively. It is worth noticing that iTb-NB is always among the top-ranked classifiers when increasing the percentage of perturbed data. When it is not ranked first, it provides values of the performance metrics that are always not far from those of the best-performing classifier.

The high resistance of iTb-NB is even more evident if the classifiers are compared in terms of Matthews' correlation coefficient (MCC). As is well known, MCC varies between -1 (worst classifier) and 1 (best classifier), whilst MCC= 0 indicates that the classifier is performing like a "toss-a-coin" model. Thus, if the MCC value is between 0 and 1 a classifier is "usable", otherwise, it is "useless". Figure 6 compares the different classifiers in terms of MCC by varying the percentage of perturbed data from of one unit at a time, from 1% to 50%. Results reported in Fig. 6 show that MCC of all the classifiers but not iTb-NB decreases rapidly to zero, or even -1, as long as the percentage of perturbed data increases from 33% to 50%. iTb-NB, instead, presents good values of MCC up to a percentage of perturbed data higher than 45%.

To further investigate about the good performance of iTb-NB, we compute the other performance metrics for iTb-NB only still varying the percentage of

Table 2 Benchmarking Tb-NB: performance metrics with the training-test set proportion 80–20

n		20.000					50.000					100.000				
	Classifier	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$
Learning SET 80%	Tb-NB	0.878	0.910	0.839	0.894	6.5	0.879	0.915	0.833	0.894	6	0.878	0.917	0.830	0.893	6.75
	Tb-NB*	(0.910)	(0.929)	(0.881)	(0.925)	(6.25)	(0.911)	(0.930)	(0.882)	(0.926)	(6)	(0.911)	(0.930)	(0.882)	(0.926)	(6.25)
	ITB-NB	0.910	0.900	0.928	0.925	5	0.919	0.917	0.924	0.932	3.5	0.922	0.922	0.922	0.934	3
	ITB-NB*	(0.912)	(0.901)	(0.932)	(0.930)	(5)	(0.922)	(0.918)	(0.929)	(0.937)	(4)	(0.925)	(0.923)	(0.928)	(0.939)	(3.75)
	NB(KLAR)	0.866	0.938	0.790	0.878	6.5	0.867	0.937	0.793	0.879	7	0.866	0.936	0.793	0.878	7.25
	NB(KLAR)*	(0.901)	(0.946)	(0.844)	(0.915)	(6.75)	(0.899)	(0.945)	(0.840)	(0.913)	(6.50)	(0.900)	(0.945)	(0.841)	(0.914)	(6.5)
	NB(E1071)	0.866	0.938	0.790	0.878	6.5	0.867	0.937	0.793	0.879	7	0.867	0.936	0.793	0.879	6.75
	NB(E1071)*	(0.901)	(0.946)	(0.844)	(0.915)	(6.75)	(0.899)	(0.945)	(0.840)	(0.913)	(6.50)	(0.900)	(0.945)	(0.841)	(0.914)	(6.5)
	RF	**0.939**	0.945	0.930	**0.948**	**1.50**	**0.940**	**0.948**	**0.928**	**0.948**	**1.00**	**0.939**	**0.949**	**0.925**	**0.947**	**1.00**
	RF*	(**0.958**)	(**0.963**)	(**0.950**)	(**0.965**)	(**1.00**)	(**0.946**)	(0.953)	(**0.937**)	(**0.954**)	(**1.25**)	(**0.946**)	(**0.955**)	(**0.932**)	(**0.953**)	(**1.00**)
	SVM	0.858	0.929	0.784	0.871	8.25	0.874	0.923	0.816	0.888	6.25	0.885	0.920	0.841	0.899	5.50
	SVM*	(0.878)	(0.940)	(0.803)	(0.893)	(8.25)	(0.891)	(0.940)	(0.829)	(0.906)	(8.25)	(0.899)	(0.939)	(0.846)	(0.913)	(7.75)
	CART	0.780	0.890	0.684	0.790	10	0.777	0.891	0.680	0.786	10	0.776	0.893	0.678	0.785	10
	CART*	(0.808)	(0.869)	(0.734)	(0.834)	(10)	(0.809)	(0.869)	(0.734)	(0.835)	(10.00)	(0.810)	(0.875)	(0.731)	(0.835)	(10)
	LDA	0.903	0.920	0.881	0.917	5.5	0.902	0.919	0.878	0.915	4.75	0.901	0.919	0.877	0.915	5
	LDA*	(0.916)	(0.939)	(0.883)	(0.929)	(5)	(0.914)	(0.939)	(0.879)	(0.928)	(5.50)	(0.914)	(0.939)	(0.878)	(0.927)	(5.25)
	LOG	0.922	**0.960**	**0.960**	0.933	1.75	0.875	0.926	0.815	0.886	6.25	0.873	0.925	0.814	0.885	6.25
	LOG*	(0.922)	(0.960)	(0.872)	(0.933)	(3.50)	(0.916)	(**0.957**)	(0.862)	(0.928)	(3.75)	(0.918)	(0.937)	(0.891)	(0.932)	(4.75)
	NN	0.922	0.937	0.901	0.932	3.50	0.911	0.921	0.889	0.923	3.25	0.908	0.924	0.887	0.921	3.5
	NN*	(0.935)	(0.951)	(0.910)	(0.945)	(2.50)	(0.923)	(0.942)	(0.896)	(0.936)	(3.25)	(0.922)	(0.940)	(0.895)	(0.934)	(3.25)

Table 2 (continued)

n		20.000					50.000					100.000				
	Classifier	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$
TEST SET 20%	Tb-NB	0.867	0.926	0.804	0.880	5.50	0.869	0.929	0.804	0.882	6	0.867	0.931	0.800	0.880	6
	Tb-NB*	(0.859)	(0.941)	(0.780)	(0.868)	(5.25)	(0.840)	(**0.945**)	(0.751)	(0.845)	(5.50)	(0.755)	(0.948)	(0.667)	(0.709)	(6)
	ITB-NB	0.875	0.915	0.827	0.890	5	0.877	0.918	0.828	0.892	5	0.877	0.920	0.826	0.891	5.75
	ITB-NB*	(0.873)	(0.931)	(0.810)	(0.886)	(5)	(0.873)	(0.935)	(0.808)	(0.886)	(5.25)	(0.874)	(0.934)	(0.809)	(0.886)	(3)
	NB(KLAR)	0.862	**0.934**	0.787	0.874	6	0.865	**0.936**	0.790	0.878	6.75	0.867	**0.936**	0.793	0.879	6.5
	NB(KLAR)*	(0.853)	(**0.943**)	(0.767)	(0.863)	(6)	(0.837)	(0.941)	(0.747)	(0.841)	(7)	(0.741)	(**0.951**)	(0.652)	(0.686)	(7.25)
	NB(E1071)	0.862	**0.934**	0.787	0.874	6	0.865	**0.936**	0.790	0.878	6.75	0.866	**0.936**	0.793	0.878	7
	NB(E1071)*	(0.853)	(**0.943**)	(0.767)	(0.863)	(6)	(0.837)	(0.941)	(0.747)	(0.841)	(7)	(0.741)	(**0.951**)	(0.652)	(0.686)	(7.25)
	RF	0.878	0.876	**0.882**	0.898	4.75	0.891	0.895	**0.886**	0.908	4.50	0.896	0.902	**0.887**	0.912	4.5
	RF*	(0.878)	(0.891)	(0.861)	(0.895)	(4.25)	(0.877)	(0.892)	(0.857)	(0.890)	(4)	(0.846)	(0.884)	(0.811)	(0.848)	(4.50)
	SVM	0.855	0.926	0.780	0.868	7.5	0.872	0.922	0.814	0.887	5.50	0.884	0.919	0.839	0.898	5.25
	SVM*	(0.842)	(0.940)	(0.752)	(0.850)	(7.75)	(0.845)	(0.940)	(0.759)	(0.852)	(5.50)	(0.793)	(0.930)	(0.710)	(0.771)	(6)
	CART	0.776	0.887	0.680	0.786	9.75	0.776	0.890	0.679	0.786	10	0.776	0.894	0.678	0.785	10
	CART*	(0.786)	(0.873)	(0.703)	(0.800)	(10)	(0.779)	(0.876)	(0.694)	(0.788)	(10)	(0.736)	(0.903)	(0.654)	(0.694)	(8.75)
	LDA	0.890	0.908	0.865	0.905	3.75	0.897	0.914	0.872	0.911	4.50	0.898	0.917	0.874	0.912	4.5
	LDA*	(0.881)	(0.926)	(0.828)	(0.893)	(4.5)	(0.875)	(0.930)	(0.816)	(0.886)	(4.75)	(0.824)	(0.919)	(0.754)	(0.815)	(4.75)

Table 2 (continued)

n		20.000					50.000					100.000				
	Classifier	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$
	LOG	**0.891**	0.908	0.869	**0.907**	**3**	**0.901**	0.916	0.879	**0.915**	3.25	**0.903**	0.919	0.881	0.916	3.25
	LOG*	(0.881)	(0.921)	(0.833)	(0.894)	(4)	(0.878)	(0.926)	(0.825)	(0.890)	(4)	(0.827)	(0.916)	(0.760)	(0.820)	(5.75)
	NN	0.890	0.909	0.865	0.906	3.75	**0.901**	0.918	0.877	**0.915**	**2.75**	**0.903**	0.920	0.880	**0.917**	**2.25**
	NN*	(**0.905**)	(0.926)	(**0.874**)	(**0.920**)	(**2.25**)	(**0.916**)	(0.936)	(**0.887**)	(**0.929**)	(**2**)	(**0.915**)	(0.934)	(**0.888**)	(**0.929**)	(**1.75**)

ACC = Accuracy; TPR = True Positive Rate; TNR = True Negative Rate; F1 = F1-score; $\overline{rk}$: average rank of the performance metrics ACC, TPR, TNR, and F1. Values in parenthesis are the performance metrics obtained considering reviews including comments with more than 3 words only. The corresponding classifier is marked with a "*". Best values of performance metrics are reported in bold

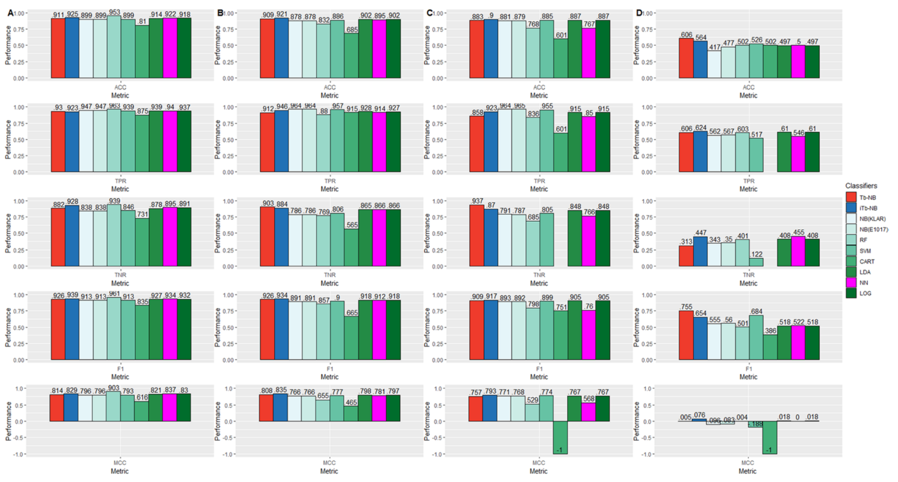


Fig. 5 Classifiers' resistance. Source: Table 7

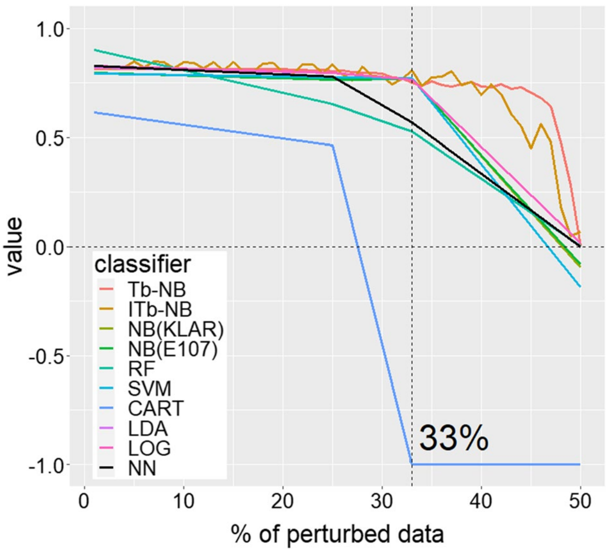


Fig. 6 Classifiers' resistance - MCC values over perturbed data percentage variation

perturbed data from one unit at a time, from 1% to 50%. In this case, besides the previously computed performance metrics, we also consider the BookMaker informedness (BM) and MarKedness (MK) (Chicco et al. 2021). Results reported in Fig. 7 show that iTb-NB (likewise Tb-NB) is resistant to noise injection with respect to all the considered classification performance metrics.

Results of the computational efficiency tests (Sect. 5.3) in terms of average training and predicting time are reported in Table 3.

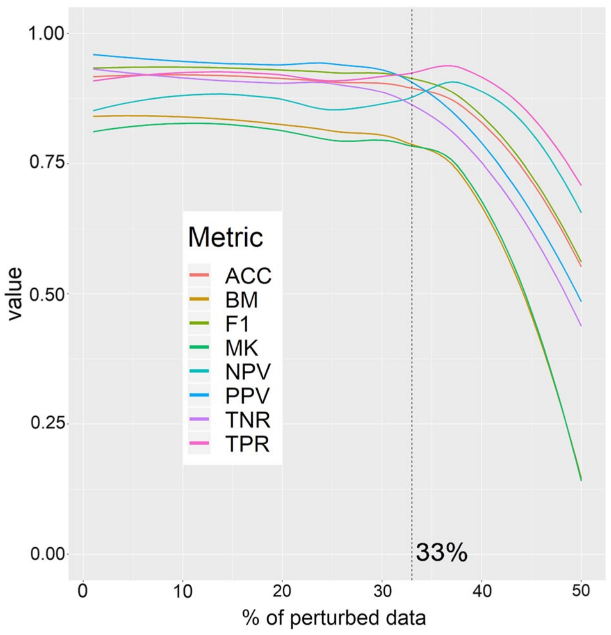


Fig. 7 iterative Threshold-based NB performance indicators

Table 3 Computational efficiency. Average training and predicting time (in seconds) of the considered classifiers for a training set of 50 observations and a test set of the same size (100 experiments)

Model	Training time	Predicting time	Training time/ Tb-NB	Predict-ing time/ Tb-NB
Tb-NB	**5.273**	**0.371**	1.000	1.000
iTb-NB	**5.434**	**0.371**	1.030	1.000
NB(KLAR)	26.999	106.608	5.120	287.654
NB(E1071)	26.999	106.608	5.120	287.654
RF	64.876	9.718	12.303	26.222
SVM	35.019	24.329	6.641	65.645
CART	11.177	4.162	2.120	11.229
LDA	8.418	0.779	1.549	2.101
LOG	102.026	30.260	19.348	81.650
NN	**0.352**	**0.055**	0.067	0.148

Results reported in Table 3 demonstrate that Tb-NB is considerably quicker both in training and predicting time compared to the others. Furthermore iTb-NB have exactly the same predicting time of Tb-NB whilst the training time is almost the same. In particular:

(a) iTb-NB and Tb-NB are ~288 times quicker in prediction, and ~5 times quicker in training, than the standard Naïve Bayes;
(b) iTb-NB and Tb-NB are ~19 times quicker in training and ~82 times quicker in prediction than logistic regression (LOG);
(c) iTb-NB and Tb-NB are ~12 times quicker in training and ~26 times quicker in prediction than random forest (RF);
(d) Besides Tb-NB, iTb-NB, LDA and CART are the less computationally demanding classifiers. However, it is worth recalling that CART is very sensitive to outliers, as well as it is most of the time the less accurate and resistant classifiers among the set of the compared classifiers. As for LDA, it works well only when the initial assumptions (Gaussian distribution for each class, linear boundary, classes with the same variance, etc.) are met.

6 Concluding remarks

People are increasingly paying attention to user-generated evaluations before buying or subscribing to a certain product or service. To get an idea of a specific product or a manufacturer in general, users read and are influenced by what others, before them, have experienced towards that product or company. Because of the enormous growth of Internet travel and tourist apps, this phenomenon is also affecting the tourism sector. In fact, travel information, particularly hotel evaluations, is one of the most popular forms of internet reviews, as reading other people's evaluations can help reduce the (sometimes unmanageable) amount of choices available (Mauri and Minazzi 2013).

From the hotel (or other type of accommodation) point of view, user evaluations are able to improve the visibility of small hospitality companies that cannot afford large advertising efforts. Since, as previously stated, people may become confused while reading hundreds of online evaluations one by one, sentiment analysis and categorization of evaluations into positive or negative have recently piqued the interest of scholars. Using this valuable tool people, but also managers, might be able to select a specific type of review, obtain a numerical evaluation of a text, and so on, in order to be able to refine choices or rethink aspects such as brand or product development.

Basically, the above-mentioned ones are among the main reasons behind the development of the iterative Threshold-based Naïve Bayes (iTb-NB) classifier. We presented a refined version of the originally implemented Threshold-based Naïve Bayes (iTb-NB) classifier (Romano et al. 2023), resulting in a versatile, and entirely data-driven classifier capable of classifying text as positive or negative based on a decision rule derived from a single threshold value computed from data. The iTb-NB classifier is fully nonparametric, as no a-priori distribution of covariates across classes is assumed, so that the results depend only on the words included in the text.

The data used to evaluate the iTb-NB performance was downloaded from Booking.com, and texts are already divided at the origin into positive and negative comments, but iTb-NB is not limited to such a case since any amount or type of text can be evaluated. We also presented a case study, namely the analysis of the reviews of

619 hotels based in Sardinia (Italy), to evaluate the performance of the classifier in a real-world application. iTb-NB has proven to be a classifier capable of competing with the many of the most popular classifiers and its performance improved that of Tb-NB in almost all the metrics used for the evaluation. Furthermore, we showed that iTb-NB is capable to provide good results even with perturbed data. Using different amounts of deliberately altered data the classifier consistently rated among the top-ranked models. Last but not least, we demonstrated that iTb-NB is also more computationally efficient than other common classifiers used for sentiment analysis.

Despite of the fact that Tb-NB classifier performs well, iTb-NB improves the performance in classifying a comment as positive or negative. What actually is even more interesting, compared to other approaches, is the versatile nature of the values produced by the scoring function $\Lambda(w_k)$ that can be aggregated together based on some specific criteria. The criterion that iTb-NB utilizes to merge values of $\Lambda(w_k)$ when classifying the out-of-fold observations is, for a given set of words included in a comment c_j ($c_j \in \mathcal{W}$), the aggregation of the values of $\Lambda(w_k)$ checking if w_k belongs (or not) to a positive comment (c_j^+) and/or to a negative comment (c_j^-) included in a review r_j. Thus, the main driver of this aggregation criterion is the out-of-fold prediction accuracy. Such a procedure, called "Post-hoc Analysis" and used for Tb-NB in Romano et al. (2023), is applicable for iTb-NB as well. Focusing on the Booking.com dataset, the iTb-NB output allows the user to assess several aspects of the service and therefore evaluate its strengths and flaws. An hotel manager can exploit the iTb-NB output in a post-hoc analysis to assess service quality and customer satisfaction levels over time, or in different locations.

Appendix

See Tables 4, 5, 6 and 7.

Table 4 Performance metrics on raw data using 5-fold cross validation

Classifier	ACC	Sensitivity	Specificity	F1	MCC
Tb-NB	0.911	0.929	0.883	0.926	0.813
iTb-NB	**0.925**	0.923	**0.928**	**0.939**	**0.829**
LOG	0.850	0.884	0.468	0.877	0.361
RF	0.811	0.873	0.409	0.849	0.303
NB(E1071)	0.806	0.804	0.611	0.834	0.390
NB(KLAR)	0.806	0.804	0.611	0.834	0.390
CART	0.768	0.842	0.413	0.815	0.272
LDA	0.764	0.860	0.359	0.816	0.246
SVM	0.793	0.929	0.710	0.771	0.621
NN	0.916	**0.936**	0.888	0.930	0.826
Average	0.835	0.878	0.628	0.859	0.505

ACC = Accuracy; F1 = F1-score; MCC = Matthews Correlation Coefficient

Table 5 Table 2 **bis**: Benchmarking Tb-NB: performance metrics with the training-test set proportion 50–50

n		20.000					50.000					100.000				
	Classifier	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$
Learning SET 50%	Tb-NB	0.878	0.907	0.841	0.894	6.50	0.878	0.912	0.836	0.894	5.75	0.878	0.915	0.833	0.893	6
	Tb-NB*	(0.909)	(0.922)	(0.881)	(0.925)	(6.00)	(0.910)	(0.929)	(0.882)	(0.926)	(5.75)	(0.910)	(0.929)	(0.881)	(0.926)	(5.75)
	ITB-NB	0.908	0.896	0.928	0.923	5	0.915	0.911	0.924	0.929	4	0.918	0.917	0.922	0.931	3.50
	ITB-NB*	(0.905)	(0.891)	(0.934)	(0.925)	(5.75)	(0.915)	(0.908)	(0.930)	(0.932)	(4.25)	(0.919)	(0.913)	(0.930)	(0.935)	(4.25)
	NB(KLAR)	0.868	0.938	0.793	0.880	7	0.865	0.937	0.788	0.877	6.75	0.864	0.937	0.788	0.876	7.25
	NB(KLAR)*	(0.898)	(0.946)	(0.837)	(0.911)	(7.25)	(0.896)	(0.945)	(0.833)	(0.909)	(7)	(0.896)	(0.945)	(0.834)	(0.910)	(7)
	NB(E1071)	0.869	0.938	0.794	0.881	6.50	0.865	0.937	0.788	0.877	6.75	0.865	0.934	0.792	0.876	6.75
	NB(E1071)*	(0.902)	(0.947)	(0.844)	(0.916)	(6.25)	(0.900)	(0.946)	(0.840)	(0.914)	(6)	(0.900)	(0.946)	(0.841)	(0.915)	(6)
	RF	**0.939**	0.943	0.933	**0.948**	**1.75**	**0.939**	**0.946**	**0.930**	**0.948**	**1.00**	**0.939**	**0.946**	**0.927**	**0.947**	**1.00**
	RF*	(0.958)	(0.963)	(**0.950**)	(0.965)	(1.75)	(**0.947**)	(0.953)	(**0.937**)	(**0.954**)	(**1.00**)	(**0.946**)	(**0.955**)	(**0.933**)	(**0.954**)	(**1.00**)
	SVM	0.851	0.930	0.771	0.863	8.25	0.865	0.926	0.896	0.878	7	0.876	0.924	0.821	0.889	5.75
	SVM*	(0.870)	(0.940)	(0.789)	(0.885)	(8.25)	(0.883)	(0.940)	(0.815)	(0.898)	(8.25)	(0.891)	(0.939)	(0.832)	(0.906)	(8.25)
	CART	0.781	0.890	0.687	0.792	10	0.777	0.891	0.681	0.787	10	0.777	0.893	0.679	0.785	10
	CART*	(0.808)	(0.870)	(0.733)	(0.834)	(10.00)	(0.809)	(0.870)	(0.733)	(0.835)	(10)	(0.810)	(0.876)	(0.731)	(0.834)	(10)
	LDA	0.906	0.922	0.883	0.919	5.50	0.902	0.919	0.879	0.916	4.75	0.902	0.919	0.878	0.915	4.75
	LDA*	(0.916)	(0.936)	(0.887)	(0.929)	(4.50)	(0.914)	(0.936)	(0.883)	(0.928)	(4.75)	(0.913)	(0.936)	(0.881)	(0.927)	(4.75)
	LOG	0.914	**0.952**	**0.951**	0.926	2	0.871	0.923	0.813	0.884	6	0.869	0.922	0.811	0.882	6.75
	LOG*	(0.916)	(0.953)	(0.865)	(0.928)	(4)	(0.910)	(**0.950**)	(0.856)	(0.923)	(5)	(0.912)	(0.930)	(0.884)	(0.927)	(3)
	NN	0.934	0.948	0.916	0.943	2.50	0.916	0.931	0.894	0.927	3	0.910	0.925	0.889	0.922	3.25
	NN*	(**0.960**)	(**0.974**)	(0.941)	(**0.967**)	(**1.25**)	(0.927)	(0.944)	(0.901)	(0.939)	(3)	(0.924)	(0.942)	(0.887)	(0.936)	(5)

Table 5 (continued)

n		20.000					50.000					100.000				
	Classifier	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$
Test SET 50%	Tb-NB	0.870	0.922	0.813	0.883	5.50	0.870	0.924	0.812	0.883	5.50	0.869	0.926	0.808	0.882	6
	Tb-NB*	(0.861)	(0.939)	(0.784)	(0.872)	(5.50)	(0.842)	(**0.943**)	(0.755)	(0.848)	(4.75)	(0.757)	(0.946)	(0.671)	(0.713)	(4.75)
	ITB-NB	0.876	0.911	0.834	0.891	4.75	0.877	0.913	0.834	0.892	5.25	0.876	0.914	0.832	0.891	5.50
	ITB-NB*	(0.873)	(0.930)	(0.811)	(0.886)	(4.75)	(0.873)	(0.934)	(0.809)	(0.886)	(4)	(0.873)	(0.933)	(0.810)	(0.886)	(4)
	NB(KLAR)	0.863	**0.932**	0.791	0.876	6.25	0.864	**0.934**	0.790	0.876	6.50	0.865	**0.935**	0.792	0.877	6.50
	NB(KLAR)*	(0.850)	(**0.942**)	(0.761)	(0.861)	(6.25)	(0.834)	(0.940)	(0.742)	(0.840)	(7.25)	(0.738)	(**0.950**)	(0.647)	(0.684)	(7.25)
	NB(E1071)	0.864	**0.932**	0.792	0.876	5.75	0.864	**0.934**	0.790	0.876	6.50	0.865	0.934	0.792	0.876	7
	NB(E1071)*	(0.854)	(0.941)	(0.769)	(0.864)	(5.75)	(0.838)	(0.939)	(0.750)	(0.843)	(6.50)	(0.742)	(0.949)	(0.655)	(0.688)	(6.50)
	RF	0.875	0.868	**0.886**	0.896	5	0.885	0.884	**0.887**	0.903	4.75	0.890	0.891	**0.888**	0.907	4.75
	RF*	(0.869)	(0.880)	(**0.853**)	(0.888)	(5)	(0.867)	(0.881)	(0.849)	(**0.883**)	(5.25)	(0.837)	(0.873)	(0.803)	(0.841)	(5.25)
	SVM	0.847	0.927	0.767	0.870	7.50	0.863	0.925	0.794	0.876	6.50	0.874	0.922	0.819	0.888	5.50
	SVM*	(0.834)	(0.940)	(0.739)	(0.842)	(7.50)	(0.837)	(0.940)	(0.746)	(0.843)	(6.50)	(0.785)	(0.930)	(0.697)	(0.763)	(6.50)
	CART	0.777	0.886	0.683	0.788	9.75	0.776	0.889	0.679	0.786	9.75	0.776	0.892	0.679	0.784	9.75
	CART*	(0.787)	(0.874)	(0.703)	(0.802)	(10)	(0.780)	(0.877)	(0.694)	(0.790)	(10)	(0.737)	(0.904)	(0.653)	(0.696)	(10)
	LDA	0.882	0.898	0.859	0.899	3.75	0.894	0.911	0.869	0.908	4.50	0.895	0.913	0.871	0.910	4.50
	LDA*	(0.880)	(0.918)	(0.833)	(0.895)	(3.25)	(0.875)	(0.922)	(0.821)	(0.888)	(3.50)	(0.823)	(0.912)	(0.759)	(0.817)	(3.50)

Table 5 (continued)

n	20.000					50.000					100.000				
Classifier	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$
LOG	**0.883**	0.900	0.860	**0.900**	**2.50**	**0.897**	0.913	0.876	**0.912**	3	0.899	0.915	0.878	0.914	3.25
LOG*	(0.875)	(0.914)	(0.827)	(0.889)	(4.25)	(0.872)	(0.919)	(0.818)	(0.885)	(5)	(0.802)	(0.903)	(0.737)	(0.782)	(5)
NN	0.880	0.899	0.854	0.897	4.25	**0.897**	0.915	0.873	**0.912**	**2.75**	**0.902**	0.918	0.880	**0.916**	**2.25**
NN*	(**0.889**)	(0.913)	(**0.853**)	(**0.907**)	(**2.75**)	(**0.911**)	(0.932)	(**0.881**)	(**0.926**)	(**2.25**)	(**0.914**)	(0.934)	(**0.885**)	(**0.928**)	(**1.75**)

ACC = Accuracy; TPR = True Positive Rate; TNR = True Negative Rate; F1 = F1-score; $\overline{rk}$: average rank of the performance metrics ACC, TPR, TNR, and F1. Values in parenthesis are the performance metrics obtained considering reviews including comments with more than 3 words only. The corresponding classifier is marked with a “*". Best values of performance metrics are reported in bold

Table 6 Table 2 **ter**: Benchmarking Tb-NB: performance metrics with the training-test set proportion 67–33

n		20.000					50.000					100.000				
	Classifier	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$
LEARNING SET 67%	Tb-NB	0.878	0.908	0.840	0.894	6.50	0.878	0.914	0.834	0.894	6	0.878	0.914	0.834	0.894	6
	Tb-NB*	(0.910)	(0.929)	(0.881)	(0.925)	(6)	(0.911)	(0.930)	(0.882)	(0.926)	(5.75)	(0.911)	(0.930)	(0.882)	(0.926)	(6.25)
	ITB-NB	0.909	0.900	0.926	0.925	5	0.917	0.914	0.924	0.931	3.50	0.917	0.914	0.924	0.931	3.50
	ITB-NB*	(0.909)	(0.896)	(0.933)	(0.927)	(5.50)	(0.919)	(0.913)	(0.930)	(0.935)	(4.25)	(0.922)	(0.918)	(0.929)	(0.937)	(4)
	NB(KLAR)	0.867	0.938	0.792	0.879	6.75	0.866	0.937	0.791	0.878	7	0.866	0.937	0.791	0.878	7
	NB(KLAR)*	(0.899)	(0.946)	(0.840)	(0.913)	(7.25)	(0.897)	(0.945)	(0.837)	(0.911)	(7)	(0.898)	(0.945)	(0.838)	(0.912)	(7)
	NB(E1071)	0.867	0.938	0.792	0.879	6.75	0.866	0.937	0.791	0.878	7	0.866	0.937	0.791	0.878	7
	NB(E1071)*	(0.902)	(0.947)	(0.840)	(0.915)	(6.25)	(0.899)	(0.946)	(0.840)	(0.913)	(6)	(0.900)	(0.945)	(0.841)	(0.914)	(5.75)
	RF	**0.938**	0.943	0.931	0.947	**1.50**	**0.939**	**0.947**	**0.929**	**0.948**	**1.00**	**0.939**	**0.947**	**0.929**	**0.948**	**1.00**
	RF*	(**0.958**)	(**0.963**)	(**0.950**)	(**0.965**)	(**1.00**)	(**0.947**)	(0.953)	(**0.937**)	(**0.954**)	(**1.25**)	(**0.946**)	(**0.955**)	(**0.932**)	(**0.955**)	(**1.00**)
	SVM	0.856	0.929	0.779	0.868	8.25	0.869	0.925	0.806	0.883	6.75	0.869	0.925	0.806	0.883	6.75
	SVM*	(0.874)	(0.940)	(0.796)	(0.889)	(8.25)	(0.887)	(0.940)	(0.822)	(0.902)	(8.25)	(0.895)	(0.939)	(0.839)	(0.910)	(7.75)
	CART	0.780	0.889	0.685	0.791	10	0.777	0.891	0.680	0.787	10	0.776	0.890	0.679	0.786	10
	CART*	(0.808)	(0.869)	(0.733)	(0.834)	(10)	(0.809)	(0.870)	(0.734)	(0.835)	(10)	(0.810)	(0.876)	(0.731)	(0.835)	(10)
	LDA	0.904	0.920	0.882	0.918	5.50	0.902	0.919	0.879	0.916	4.75	0.902	0.919	0.879	0.916	4.75
	LDA*	(0.916)	(0.938)	(0.885)	(0.929)	(4.75)	(0.914)	(0.938)	(0.881)	(0.928)	(5)	(0.914)	(0.938)	(0.879)	(0.927)	(5.50)
	LOG	0.920	**0.958**	**0.957**	0.931	2.75	0.873	0.925	0.814	0.885	5.75	0.873	0.925	0.814	0.885	5.75
	LOG*	(0.919)	(0.957)	(0.868)	(0.931)	(3.75)	(0.913)	(**0.954**)	(0.859)	(0.926)	(4.50)	(0.915)	(0.934)	(0.888)	(0.930)	(4.75)
	NN	0.926	0.939	0.907	0.936	2	0.913	0.929	0.891	0.925	3.25	0.909	0.925	0.887	0.922	3.25
	NN*	(0.942)	(0.957)	(0.919)	(**0.951**)	(2.25)	(0.925)	(0.943)	(0.899)	(0.937)	(3)	(0.922)	(0.941)	(0.895)	(0.935)	(3)

Table 6 (continued)

n		20.000					50.000					100.000				
	Classifier	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$
TEST SET 33%	Tb-NB	0.870	0.923	0.811	0.884	5.50	0.869	0.926	0.808	0.883	5.25	0.869	0.926	0.808	0.883	5.25
	Tb-NB*	(0.860)	(0.940)	(0.782)	(0.870)	(5.50)	(0.841)	(**0.945**)	(0.753)	(0.846)	(5)	(0.756)	(0.947)	(0.669)	(0.711)	(6)
	ITB-NB	0.876	0.913	0.833	0.892	5	0.877	0.916	0.831	0.892	5.25	0.877	0.916	0.831	0.892	5.25
	ITB-NB*	(0.873)	(0.930)	(0.811)	(0.886)	(5)	(0.873)	(0.934)	(0.808)	(0.886)	(4.75)	(0.873)	(0.933)	(0.809)	(0.886)	(2.75)
	NB(KLAR)	0.865	**0.935**	0.791	0.878	6	0.865	**0.935**	0.790	0.877	6.75	0.865	**0.935**	0.790	0.877	6.75
	NB(KLAR)*	(0.852)	(**0.943**)	(0.764)	(0.862)	(6.25)	(0.835)	(0.940)	(0.744)	(0.841)	(7.25)	(0.740)	(**0.950**)	(0.650)	(0.685)	(7.50)
	NB(E1071)	0.865	**0.935**	0.791	0.878	6	0.865	**0.935**	0.790	0.877	6.75	0.865	**0.935**	0.790	0.877	6.75
	NB(E1071)*	(0.854)	(0.942)	(0.768)	(0.863)	(5.75)	(0.837)	(0.940)	(0.748)	(0.842)	(6.75)	(0.742)	(**0.950**)	(0.654)	(0.687)	(6.75)
	RF	0.882	0.878	**0.888**	0.902	4.75	0.888	0.889	**0.886**	0.906	4.75	0.888	0.889	**0.886**	0.906	4.75
	RF*	(0.873)	(0.886)	(0.857)	(0.892)	(4.75)	(0.872)	(0.886)	(0.853)	(0.887)	(5)	(0.842)	(0.879)	(0.807)	(0.844)	(4.75)
	SVM	0.852	0.926	0.774	0.865	7.50	0.868	0.923	0.804	0.881	6.25	0.868	0.923	0.804	0.881	6.25
	SVM*	(0.838)	(0.940)	(0.745)	(0.846)	(7.50)	(0.841)	(0.940)	(0.753)	(0.848)	(6)	(0.789)	(0.930)	(0.703)	(0.767)	(6)
	CART	0.777	0.887	0.681	0.788	9.75	0.776	0.890	0.679	0.786	9.75	0.776	0.890	0.679	0.786	9.75
	CART*	(0.787)	(0.873)	(0.703)	(0.801)	(10)	(0.779)	(0.877)	(0.694)	(0.789)	(10)	(0.737)	(0.904)	(0.654)	(0.695)	(9)
	LDA	0.888	0.906	0.862	0.904	3.25	0.895	0.913	0.871	0.910	4.50	0.895	0.913	0.871	0.910	4.50

Table 6 (continued)

n		20.000					50.000					100.000				
	Classifier	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$	ACC	TPR	TNR	F1	$\overline{rk}$
	LDA*	(0.881)	(0.922)	(0.831)	(0.894)	(3.25)	(0.875)	(0.926)	(0.819)	(0.887)	(4.25)	(0.824)	(0.916)	(0.756)	(0.816)	(4.75)
	LOG	**0.889**	0.906	0.866	**0.905**	**2.75**	**0.899**	0.914	0.877	**0.913**	3	0.899	0.914	0.877	0.913	3.50
	LOG*	(0.878)	(0.918)	(0.830)	(0.892)	(4.50)	(0.875)	(0.923)	(0.821)	(0.887)	(3.75)	(0.805)	(0.906)	(0.740)	(0.785)	(5.75)
	NN	0.887	0.905	0.861	0.902	4.50	**0.899**	0.917	0.875	**0.913**	**2.75**	**0.904**	0.919	0.882	**0.917**	**2.25**
	NN*	(**0.900**)	(0.922)	(**0.868**)	(**0.916**)	(**2.50**)	(**0.913**)	(0.934)	(**0.883**)	(**0.927**)	(**2.25**)	(**0.915**)	(0.934)	(**0.886**)	(**0.928**)	(**1.75**)

ACC = Accuracy; TPR = True Positive Rate; TNR = True Negative Rate; F1 = F1-score; $\overline{rk}$: average rank of the performance metrics ACC, TPR, TNR, and F1. Values in parenthesis are the performance metrics obtained considering reviews including comments with more than 3 words only. The corresponding classifier is marked with a "*". Best values of performance metrics are reported in bold

Table 7 Classifiers' resistance - performance metrics obtained by cross-validation for each percentage of perturbed data

Classifier	Perturbed Data(%)	ACC	TPR	TNR	F1	MCC	$\overline{rk}$
Tb-NB		0.911	0.930	0.882	0.926	0.814	6.2
iTb-NB		**0.925**	0.923	**0.928**	**0.939**	0.829	3.8
NB(KLAR)		0.899	**0.947**	0.838	0.913	0.796	6.9
NB(E1071)		0.899	**0.947**	0.838	0.913	0.796	6.9
RF		**0.953**	**0.963**	**0.939**	**0.961**	**0.903**	1.0
SVM		0.899	0.939	0.846	0.913	0.793	7.6
CART		0.810	0.875	0.731	0.835	0.616	10
LDA		0.914	0.939	0.878	0.927	0.821	5.2
LOG		0.918	0.937	0.891	0.932	**0.830**	4.4
NN	0	0.918	0.937	0.891	0.932	0.830	3
TB-NB		**0.909**	0.912	**0.903**	**0.926**	**0.808**	3.2
iTB-NB		**0.921**	0.946	**0.884**	**0.934**	**0.835**	1.8
NB(KLAR)		0.878	**0.964**	0.786	0.891	0.766	6.3
NB(E1071)		0.878	**0.964**	0.786	0.891	0.766	6.3
RF		0.832	0.880	0.769	0.857	0.655	9.2
SVM		0.886	**0.957**	0.806	0.900	0.777	5.4
CART		0.685	0.915	0.565	0.665	0.465	9.4
LDA		0.902	0.928	0.865	0.918	0.798	4
LOG		0.902	0.927	0.866	0.918	0.797	4.1
NN	25	0.895	0.914	0.866	0.912	0.781	5.3
TB-NB		0.883	0.858	**0.937**	**0.909**	0.757	4.4
iTB-NB		**0.900**	0.923	**0.870**	**0.917**	**0.793**	1.8
NB(KLAR)		0.881	**0.964**	0.791	0.893	0.771	4.6
NB(E1071)		0.879	**0.965**	0.787	0.892	0.768	5.2
RF		0.768	0.836	0.685	0.798	0.529	8.6
SVM		0.885	0.955	0.805	0.899	**0.774**	3.8
CART		0.601	0.601	NaN	0.751	−1.000	10
LDA		**0.887**	0.915	0.848	0.905	0.767	4.1
LOG		**0.887**	0.915	0.848	0.905	0.767	4.1
NN	33	0.767	0.850	0.766	0.760	0.568	8.4
TB-NB		**0.606**	0.606	0.313	**0.755**	**0.005**	3.6
iTB-NB		**0.564**	**0.624**	**0.447**	0.654	**0.076**	1.8
NB(KLAR)		0.417	0.562	0.343	0.555	−0.096	7.4
NB(E1071)		0.477	0.567	0.350	0.560	−0.083	6.4
RF		0.502	0.603	0.401	0.501	0.004	5.7
SVM		0.526	0.517	0.122	**0.684**	−0.188	6.4
CART		0.502	NaN	NaN	0.386	−1.000	8.9
LDA		0.497	**0.610**	**0.408**	0.518	0.018	4.7
LOG		0.497	**0.610**	**0.408**	0.518	0.018	4.7
NN	50	0.500	0.546	0.455	0.522	0.000	5.4

ACC = Accuracy; TPR = True Positive Rate; TNR = True Negative Rate; F1 = F1-score; MCC = Matthews' correlation coefficient; $\overline{rk}$ = average rank of the performance metrics ACC, TPR, TNR, F1 and MCC. Best values of performance metrics are reported in bold

Funding Open access funding provided by Università degli Studi di Cagliari within the CRUI-CARE Agreement.

Declarations

Conflict of interest The authors declare no competing interests.

References

Alaei AR, Becken S, Stantic B (2019) Sentiment analysis in tourism: capitalizing on big data. Journal of Travel Research 58(2):175–191

Chai C (2019) Text mining in survey data. Survey Practice 12:1–13. https://doi.org/10.1017/S1351324920000534

Chicco D, Jurman G (2020) The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. BMC Genomics 21(6). https://doi.org/10.1186/s12864-019-6413-7

Chicco D, Tötsch N, Jurman G (2021) The Matthews correlation coefficient (MCC) is more reliable than balanced accuracy, bookmaker informedness, and markedness in two-class confusion matrix evaluation. BioData Mining 14(13):1–22. https://doi.org/10.1186/s13040-021-00244-z

Denecke K, Deng Y (2015) Sentiment analysis in medical settings: New opportunities and challenges. Artificial intelligence in medicine 64(1):17–27

Dey L, Chakraborty S, Biswas A, et al (2016) Sentiment analysis of review datasets using naive bayes and k-nn classifier. arXiv preprint arXiv:1610.09982

Dogan N, Tanrikulu Z (2013) A comparative analysis of classification algorithms in data mining for accuracy, speed and robustness. Information Technology and Management 14(2):105–124. https://doi.org/10.1007/s10799-012-0135-8

Fang X, Zhan J (2015) Sentiment analysis using product review data. Journal of Big Data 2(1):1–14

Fritsch S, Guenther F, Wright MN (2019) neuralnet: Training of Neural Networks. https://CRAN.R-project.org/package=neuralnet, r package version 1.44.2

Karr AF, Bowen Z, Porter AA (2022) Structure of classifier boundaries: Case study for a naive bayes classifier. arXiv:ArXiv2212.04382

Laksono RA, Sungkono KR, Sarno R, et al (2019) Sentiment analysis of restaurant customer reviews on tripadvisor using naïve bayes. In: 2019 12th International Conference on Information & Communication Technology and System (ICTS). IEEE, p 49–54

Mauri AG, Minazzi R (2013) Web reviews influence on expectations and purchasing intentions of hotel potential customers. International journal of hospitality management 34:99–107

Micu A, Micu AE, Geru M et al (2017) Analyzing user sentiment in social media: Implications for online marketing strategy. Psychology & Marketing 34(12):1094–1100

Mohammad SM (2016) Sentiment analysis: Detecting valence, emotions, and other affectual states from text. In: Emotion measurement. Elsevier, p 201–237

Morante R, Blanco E (2021) Recent advances in processing negation. Natural Language Engineering 27:121–130. https://doi.org/10.1007/s10115-019-01410-w

Nguyen TH, Shirai K, Velcin J (2015) Sentiment analysis on social media for stock movement prediction. Expert Systems with Applications 42(24):9603–9611

Riedmiller M, Braun H (1993) A direct adaptive method for faster backpropagation learning: the rprop algorithm. In: IEEE International Conference on Neural Networks, pp 586–591, https://doi.org/10.1109/ICNN.1993.298623

Romano M, Zammarchi G, Conversano C (2022a) Iterative threshold-based naïve bayes classifier: an efficient tb-nb improvement. In: Book of the Short papers SIS 2022 51st Scientific Meeting of the Italian Statistical Society, pp 1656–1661

Romano M, Zammarchi G, Conversano C (2022b) Threshold-based naïve bayes classifier: Customer satisfaction evaluation. In: Short Papers IES 2022 Innovation & Society 5.0: Statistical and Economic Methodologies for Quality Assessment, pp 90–94

Romano M, Contu G, Mola F et al (2023) Threshold-based naïve bayes classifier. Advances in Data Analysis and Classification. https://doi.org/10.1007/s11634-023-00536-8

Sánchez-Franco MJ, Navarro-García A, Rondán-Cataluña FJ (2019) A naive bayes strategy for classifying customer satisfaction: A study based on online reviews of hospitality services. Journal of Business Research 101:499–506

Tavazoee F, Conversano C, Mola F (2020) Recurrent random forest for the assessment of popularity in social media. Knowledge and Information Systems 62:1847–1879. https://doi.org/10.1007/s10115-019-01410-w

Tran T, Ba H, Huynh VN (2019) Measuring hotel review sentiment: An aspect-based sentiment analysis approach. In: International Symposium on Integrated Uncertainty in Knowledge Modelling and Decision Making. Springer, p 393–405

Vidya NA, Fanany MI, Budi I (2015) Twitter sentiment to analyze net brand reputation of mobile phone providers. Procedia Computer Science 72:519–526

Webb GI, Keogh E, Miikkulainen R (2010) Naïve bayes. Encyclopedia of machine learning 15:713–714

Wilson T, Wiebe J, Hoffmann P (2005) Recognizing contextual polarity in phrase-level sentiment analysis. In: Proceedings of human language technology conference and conference on empirical methods in natural language processing, pp 347–354

Xu F, Pan Z, Xia R (2020) E-commerce product review sentiment classification based on a naïve bayes continuous learning framework. Information Processing & Management 57(5):102,221

Yadollahi A, Shahraki AG, Zaiane OR (2017) Current state of text sentiment analysis from opinion to emotion mining. ACM Computing Surveys (CSUR) 50(2):1–33

Yu LC, Wang J, Lai KR et al (2018) Refining word embeddings using intensity scores for sentiment analysis. IEEE/ACM Transactions on Audio, Speech, and Language Processing 26(3):671–681. https://doi.org/10.1109/TASLP.2017.2788182

Yue L, Chen W, Li X et al (2019) A survey of sentiment analysis in social media. Knowledge and Information Systems 60(2):617–663

Zammarchi G, Mola F, Conversano C (2023) Using sentiment analysis to evaluate the impact of the covid-19 outbreak on italy's country reputation and stock market performance. Statistical methods & applications pp 1–22

Zhang H, Jiang L, Yu L (2021) Attribute and instance weighted naive bayes. Pattern Recognition 111(107):674. https://doi.org/10.1016/j.patcog.2020.107674

Zhang H, Jiang L, Webb GI (2023) Rigorous non-disjoint discretization for naive bayes. Pattern Recognition 140(109):554. https://doi.org/10.1016/j.patcog.2023.109554