BOOKER PREDICTION FROM REQUESTS FOR QUOTATION VIA MACHINE LEARNING TECHNIQUES

Samuel RUNGGALDIER, MSc
Free University of Bozen-Bolzano,
Faculty of Computer Science
Piazza Domenicani, 3, Bolzano, Italy
E-mail: Samuel.Runggaldier@stud-inf.unibz.it

Gabriele SOTTOCORNOLA, PhD, Research Assistant *(Corresponding Author)*
Free University of Bozen-Bolzano,
Faculty of Computer Science
Piazza Domenicani, 3, Bolzano, Italy
E-mail: Gabriele.Sottocornola@unibz.it

Andrea JANES, PhD, Researcher
Free University of Bozen-Bolzano,
Faculty of Computer Science
Piazza Domenicani, 3, Bolzano, Italy
E-mail: Andrea.Janes@unibz.it

Fabio STELLA, PhD, Associate Professor
University of Milano-Bicocca,
Department of Informatics, Systems and Communication
Viale Sarca, 336, Milano, Italy
E-mail: Fabio.Stella@unimib.it

Markus ZANKER, PhD, Full Professor
Free University of Bozen-Bolzano,
Faculty of Computer Science
Piazza Domenicani, 3, Bolzano, Italy
E-mail: Markus.Zanker@unibz.it
University of Klagenfurt
Faculty of Technical Sciences
Universitätsstrasse 65, Klagenfurt, Austria
E-mail: Markus.Zanker@aau.at

Abstract

Purpose – Many incoming requests for quotation usually compete for the attention of accommodation service provider staff on a daily basis, while some of them might deserve more priority than others.
Design – This research is therefore based on the correspondence history of a large booking management system that examines the features of quotation requests from aspiring guests in order to learn and predict their actual booking behavior.
Approach – In particular, we investigate the effectiveness of various machine learning techniques for predicting whether a request will turn into a booking by using features such as the length of stay, the number and type of guests, and their country of origin. Furthermore, a deeper analysis of the features involved is performed to quantify their impact on the prediction task.
Findings – We based our experimental evaluation on a large dataset of correspondence data collected from 2014 to 2019 from a 4-star hotel in the South Tyrol region of Italy. Numerical experiments were conducted to compare the performance of different classification models against the dataset. The results show a potential business advantage in prioritizing requests for proposals based on our approach. Moreover, it becomes clear that it is necessary to solve the class imbalance problem and develop a proper understanding of the domain-specific features to achieve higher precision/recall for the booking class. The investigation on feature importance also exhibits a ranking of informative features, such as the duration of the stay, the number of days prior to the request, and the source/country of the request, for making accurate booking predictions.
Originality of the research – To the best of our knowledge, this is one of the first attempts to apply and systematically harness machine learning techniques to request for quotation data in order to predict whether the request will end up in a booking.
Keywords Booking Prediction, Request for Quotation, Machine Learning, Class Imbalance Problem, Feature Importance Analysis

Original scientific paper
Received 12 May 2022
Revised 8 September 2022
31 October 2022
Accepted 4 November 2022
https://doi.org/10.20867/thm.29.1.3

INTRODUCTION

The Tourism sector has been majorly transformed by the digitization of processes (Gretzelet al., 2015) and, thus, aspiring travelers inquiring multiple hotels for quotations and picking the best offer is commonplace. Correspondence management has therefore become one of the most important tasks in hotel management. However, creating and sending offers is time-consuming causing high employee costs. The potential to optimize the sales and acquisition process is reinforced by the fact that only a small percentage of offers made actually turns into bookings. This means that most of the effort on creating and writing offers is wasted on non-booking customers. Thus, the ability to accurately predict whether a guest will book or not, given a request for quotation, can help to set priorities of the correspondence management workload and increase conversion rates (i.e., the share of customers booking an offer). Depending on the likelihood of actual bookings, appropriate decisions can be taken to increase the chances to convert a request into a booking.

Nevertheless, such a problem seems to be under-explored in current tourism-related scientific literature (Egger, 2022). Thus, we aim at bridging this research gap by proposing an experimental investigation of automated algorithms to predict the likelihood of a request for quotation turning into a booking, based on its features. We built our methodological research upon previous works on booking cancellation prediction (Antonio et al., 2017) and revenue and booking forecasting (Pan et al., 2012, Li et al., 2017). We leveraged real-world logged data to evaluate the classification effectiveness of different algorithms (i.e., tree-based, bayesian, and neural network algorithms) and techniques (i.e., class subsampling and feature selection), as well as to provide qualitative insights on customers behavior.

In particular, we collaborated with an IT company, which provided access to a data log with anonymized past interactions with their correspondence management software. The software is employed by tourism service providers such as accommodation facilities to collect, manage and reply to incoming requests for quotation in an effective and practical manner. The requests may

originate from different channels such as e-mails, phone calls or web portals. We focused our analysis on a specific hotel with a long-lasting usage of the correspondence software, namely a 4-star hotel with both summer and winter tourism.

The major goal of our research concerns the application of machine learning data-driven algorithms to predict whether a customer asking for a request for quotation will turn into a booker. Furthermore, this problem naturally suffers from a value imbalance for the class we want to predict (i.e., whether a request will convert into a booking or not). Namely, booking conversion represents a small percentage of all the requests for quotation processed by the software and included in our data log. We investigate how this problem impacts our classification performances and how to cope with it (Ali et al., 2013). For the purpose of analysis, we focus on request features (i.e., the independent variables), such as duration of stay and customer demographics like number of adults and children, in order to predict actual conversion into booking (i.e., the dependent variable). More specifically, we want to assess the effectiveness of different machine learning techniques on the booking prediction task, as well as the impact of different features, inspired by the work by Cezar & Ogüt (2016). Thus, we are guided by the following research questions:

- **RQ 1** Is it possible to predict whether a request for quotation will turn into a booking via supervised classification algorithms?
- **RQ 2** How can we address the class imbalance (i.e., the proportion of non-booking requests in the data is much higher than bookings) to improve classification performance?
- **RQ 3** Can we discover important features that mostly contribute to classification accuracy?
- **RQ 4** Which feature values best discriminate between booking and non-booking requests?

The rest of the paper is organized as follows: in Section 1 we present some related applications of machine learning techniques in the area of tourism; in Section 2 we describe the data used in experiments, the context from which they are generated, and the applied pre-processing steps; in Section 3 we describe the machine learning experiments and discuss results, before drawing conclusions.

1. RELATED WORK

The optimization of sales processes and correspondence management in the accommodation industry with the help of machine learning techniques is clearly under-investigated in scientific research (Egger, 2022). Therefore, we survey machine learning applications for similar tasks in tourism-related scientific literature. The works presented in this review, categorized by topic, are reported in Table 1.

Table 1: **Related works classified by topic**

Booking/cancellation prediction	(Egger, 2022); (Thomas et al., 2019); (Antonio et al., 2017); (Chiang et al., 2007); (Kimes & Wirtz, 2003)
Class imbalance problem	(Ali et al., 2013); (Thabtah et al., 2020); (Adil et al., 2021)
Feature analysis	(Çiftçi & Cizel, 2020); (Wong et al., 2020); (Cezar & Ogüt, 2016); (Xie & Lee, 2020); (Khatibi et al., 2020)
Revenue and booking forecasting	(Webb et al., 2020); (Fiori & Foroni, 2020); (Schwartz et al., 2016); (Pan et al., 2012); (Assaf et al., 2019); (Pan & Yang, 2017); (Chen & Wang, 2007); (Yang et al., 2015); (Li et al., 2017); (Claveria & Torra, 2014); (Höpken et al., 2017); (Lado-Sestayo & Vivel-Búa, 2018, 2019)

Booking prediction in order to manage overbooking is one of the key techniques of revenue management. After the Airline Deregulation Act of 1978 deregulated pricing policies revenue management started to play an important role in the aviation industry. However, as early as in 1966 American Airlines developed their first computer-based revenue management system (Chiang et al., 2007). Inspired by the aviation industry, the hotel industry also started to implement revenue management and prediction of cancellations (Chiang et al., 2007, Kimes & Wirtz, 2003). A related research conducted by Antonio et al., (2017) demonstrated the possibility to predict booking cancellations with an accuracy of 90% in a real-world setting. The experiments were performed using the data of four hotels in the resort region of Algarve, Portugal. Similarly to the available features in our research, they also used information such as the duration of stay, number of guests and demographic information, such as nationality, as well as characteristics of the booking process, such as the time prior to arrival of the booking itself.

In the recommendation context, specifically, for flight-based travel booking, an application of a cascaded machine learning model was proposed (Thomas et al., 2019). The goal was to select the best set of hotels to recommend to the traveler, based on the estimated probability of conversion for each candidate hotel and on analysis of the feature importance derived from flight information. Differently, machine learning was used to determine users' e-trust in adopting web-based travel intermediaries (Çiftçi & Cizel, 2020). Specifically, hierarchical linear regression was applied to investigate which factors affect tourists' e-trust perception. Similarly, Wong et al., (2020) applied partial least squares and importance-performance map analysis to identify the relationship between service quality and hotel guest satisfaction.

In the tourism domain, some effort has also been dedicated to better understand the booking conversion problem with respect

to specific browsing activity features. For instance, Cezar & Ogüt (2016) proposed an analysis on the impact of web browsing-related dependent variables, namely, review ratings (on location and service aspects), recommendation, and search rankings to the conversion of a browsing activity into a booking. Similarly, Xie & Lee (2020) investigated how informational cues displayed in an online hotel search process, including quality indicators, brand affiliation, incentives (discounted price and promotion) and position in search results, influenced consumer conversion at different stages of their browsing session. Finally, the importance of environmental and social media features in the tourism attraction prediction was studied by Khatibi et al., (2020).

Another relevant problem to be tackled in our work and in general in tourism-related machine learning classification is the one of class imbalance (Ali et al., 2013). Generally, the data points related to a positive event (i.e. booking) are much more rare than the corresponding negative event (i.e. non-booking). Thus, data-driven classifiers can struggle to correctly recognize positive instances. Many different techniques were proposed in the literature to address such a problem (Thabtah et al., 2020), for instance, Adil et al., (2021) implemented an oversampling technique to the minority class, in order to facilitate supervised classification of hotel booking cancellations. Contrary to our proposal, where instead a down-sampling of the majority class at different ratios is analyzed.

A similar topic concerns the forecasting of revenue management (Webb et al., 2020, Fiori & Foroni, 2020) and occupancy/ booking (Schwartz et al., 2016, Pan et al., 2012) by means of automated algorithms. Indeed, a large share of related work exists in the context of forecasting tourism demand. This refers, for instance, to the regression problem of predicting the number of arrivals in vacation destinations based on past collected data (Assaf et al., 2019, Pan & Yang, 2017). In (Chen & Wang, 2007) the authors compared support vector regression, with neural networks and autoregressive integrated moving average for the task of forecasting tourism demand in China in the years 1985 to 2001. Related works by Yang et al., (2015) and Li et al., (2017), combined search engines data and visitors volume to boost the accuracy of tourism demand forecasting in popular locations in China. Similar research was conducted by Claveria & Torra (2014), where autoregressive methods are compared with artificial neural networks to forecast tourism arrivals in Catalonia from 2001 to 2009. Höpken et al., (2017) exploit mining approaches (i.e., nearest neighbors and linear regression) on big data, such as web search traffic, in order to predict tourist arrivals in Are (Sweden) for the period of 2005-2012. Finally, hotel profitability forecasting was addressed by means of partial least squares and clustering techniques (Lado-Sestayo & Vivel-Búa, 2018), as well as through deep neural network regressors (Lado-Sestayo & Vivel-Búa, 2019).

2. DATA DESCRIPTION

In this section we explain the context, the structure and the format of the data used for the analysis. Moreover, we describe the software and the procedure from which the data is generated during the booking procedure.

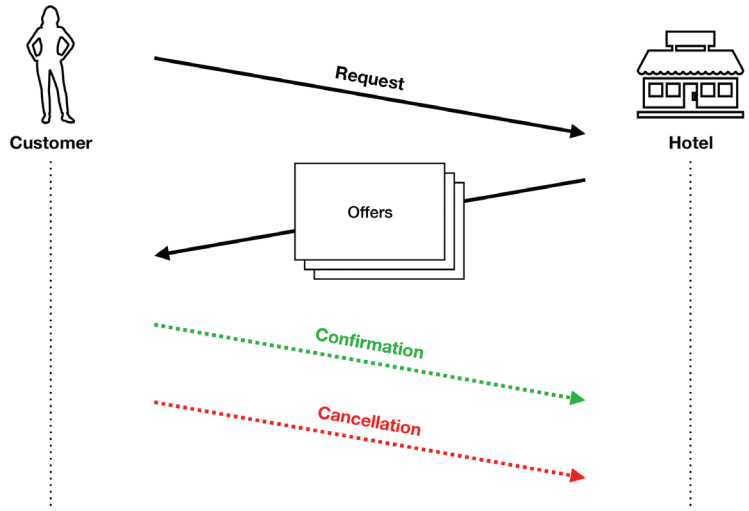
2.1. Booking Procedure

The correspondence software, from which the data for the booking prediction is derived, is a hotel management software. The software is used by a total of 92 accommodation companies. It is used by 44 3-star, 13 4-star and one 5-star hotels, by 9 guest houses, 7 garni and 18 residences. Most of them are located in South Tyrol, Italy in ski and/or hiking destinations. Its database contains log data that spans from 2014 up to the first months of 2019.

The procedure for booking a room through the correspondence software is displayed in the sequence diagram of Figure 1. The procedure for booking a room through the correspondence software consists of the following steps.
1. The guest generates a request for quotation which may come from a web portal, an e-mail or a direct call to the hotel.
2. The hotels responsible to handle the request send a letter (i.e., an e-mail) that contains one or multiple offers.
3. The guest may choose one of the received offers and confirm the reservation (i.e., she books the offer).
4. The guest may decide to cancel a booked offer, if allowed by the hotel policy.

Step 1. is the key step where most of the data used in our analysis is collected. In this step the customers provide demographic information about themselves and their fellow travelers, the period of stay, and some specific requests, described in free text. Furthermore, meta-information, such as the source of communication and the preferred type of communication, is collected. In Step 2. the receptionist of the hotel generates a set of offers to answer each specific request. The added information for each offer is related to the number of rooms, the room types, the boarding types and the price. Finally, in the optional steps 3 and 4 the status of the requests are updated based on the guests' responses.

Figure 1: **The sequence diagram of the request-booking procedure that may end up in a reservation or a cancellation**



2.2. Dataset

After a pre-processing and data cleaning phase we came up with the final dataset, used in our experiments. Table 2 lists all features included in the dataset, Table 3 summarizes the descriptive statistics for the numeric independent variables. For the categorical variables, the number of unique values in the data are: 66 for "CG_CountryCode", 30 for "CR_SourceOfBusiness", 4 for "CG_Gender" (man, woman, company, group), 3 for "CG_LanguageId" (German, Italian, English), and 2 for "CR_Season" (winter, summer).

Table 2: **Description of the independent variables extracted from the database for the classification task**

Name	Type	Description
CR_Adults	Numeric	Number of adults.
CR_Children	Numeric	Number of children.
CR_Age_0-3	Numeric	Number of children between 0 and 3 years.
CR_Age_4-10	Numeric	Number of children between 4 and 10 years.
CR_Age_11-17	Numeric	Number of children between 11 and 17 years.
CR_Season	Categorical	Season of the stay (winter or summer).
CR_Duration	Numeric	Duration of the stay in days.
CR_SourceOfBusiness	Categorical	The agent or portal from where the request came.
CG_Gender	Categorical	The gender of the enquirer. In addition to woman and man, company and group were also considered as gender.
CG_LanguageId	Categorical	The language of enquirer (German, Italian, English).
CG_CountryCode	Categorical	The country of the enquirer
L_RecallEmailSended	Boolean	Whether in addition to a HTML e-mail also a plain-text e-mail was sent
L_SendNachhakEmail	Boolean	Whether a reminder e-mail was sent after some days
L_UseWebTemplate	Boolean	Whether a web-email or normal email was sent
CWM_Message	Numeric	Number of words in the message
CR_RequestedDays BeforeArrival	Numeric	Number of days prior to arrival on which the first offer was sent.

Table 3: **Descriptive statistics for the numeric independent variables**

Name	Mean	Std	Median	Min	Max
CR_Adults	2.472	3.075	2	1	8
CR_Children	0.532	0.750	0	0	8
CR_Age_0-3	0.120	0.358	0	0	5
CR_Age_4-10	0.298	0.534	0	0	5
CR_Age_11-17	0.113	0.335	0	0	7
CR_Duration	5.501	2.909	6	1	16
CWM_Message	1.301	30.714	0	0	271
CR_RequestedDays BeforeArrival	66.199	67.747	34	1	336

The experimentation focused on the data of a specific 4-star hotel with winter and summer season targeting skiing and hiking tourists. By restricting our analysis to a single hotel with an intense and long-lasting usage history of the correspondence management software we keep the workflow of the booking process under control. On the other hand, we are aware of the fact that by constraining the analysis to a single hotel we may incur some biases (e.g. the origin country of the enquirer is geographically conditioned on the location of the hotel), which might represent a limitation of the proposed approach. We stress the fact that data were anonymized, such that it was not possible to consider personalized and historical characteristics of returning customers.

The dataset therefore contains a total of 57054 requests after pre-processing out of which 4919 ended up in a booking while 52135 did not. Thus, 8.62% of the incoming requests finally turned into bookings, i.e., belonging to the positive class, and the remaining 91.38% to the negative class. It is clear that this dataset suffers from a serious problem of class imbalance. In order to cope with this problem, class imbalance subsampling was applied and its effect analyzed.

3. EXPERIMENTS AND RESULTS

The main focus of our set of experiments is to evaluate and compare the effectiveness of various machine learning techniques applied to the prediction of booking requests. For the sake of clarity, we divided the experiments into three main sub-phases: (i) we compare a set of classification algorithms to select the best models for the task; (ii) we evaluate the effect of sub-sampling to alleviate the class imbalance problem; (iii) we apply feature selection to understand the importance of different features and their impact on the classification task. In order to ensure replicability and transparency, for each of these steps we will describe in detail how the experiments were set-up, and we present and discuss results. For all experiments the Python library scikit-learn (*https://scikit-learn.org/stable/*) was used for data processing, classification, and evaluation.

3.1. Model Selection

For the model selection task we provided a comprehensive analysis of the different classification approaches in the literature (Tan, 2005), hence comparing their classification performances over a large plethora of algorithms, each with its own peculiarity. Specifically, 16 different classifiers were tested. Out of these classifiers, the five best were selected for deeper analysis. In Appendix B, we fully compare the mean and standard deviation results for all 16 classifiers. For the sake of clarity we report and discuss just the results of the five best performing models. Namely, Random Forest and Extra Tree (Rokach & Maimon, 2014), Gaussian Naive Bayes (Murphy, 2012), Multi-layer Perceptron (MLP) (Gurney, 1997) and Support Vector Classifier (SVC) with radial basis function kernel (Schölkopf & Smola, 2001). The results were also benchmarked with two baselines, i.e., dummy classifiers with most frequent and stratified strategies. All classifiers were tested and evaluated with default parameters provided by scikit-learn library, please refer to the documentation (*https://scikit-learn.org/stable/*) for the complete list of parameters and their values.

A brief description of each of those five algorithms is provided in Appendix A, together with a formal definition of the metrics used in the evaluation.

We designed a robust evaluation protocol to test and compare the performance of each classifier. This protocol guarantees a high degree of generalization and significance of the reported results. Namely, for each model we performed 10 times a 10-fold cross-validation on the whole dataset. The data were shuffled at the beginning of each iteration, before the 10 folds were sampled. To ensure the original class distribution, we apply a stratified random sampling over the folds (Tan, 2005). The mean and standard deviation of different classification performance scores, namely, precision, recall, F1-score, and accuracy, for the 10 replications are shown in Table 4. The speed index indicates how fast each of the five classifiers is trained in comparison to the others. Please notice that one-hot encoding was applied to categorical features in order to fit every type of training

algorithm. The selected classifiers on which the analysis of the results is conducted show a diverse behavior that outperforms all others with respect to at least one metric.

Table 4: **Mean and standard deviation results for the five selected classifiers and the two baseline dummy classifiers evaluated through 10-fold cross validation**

Classifier	Speed	Measure	Precision	Recall	F1	Accuracy
Extra Tree	0.003	Mean	0.343	0.329	0.336	0.887
		Std	0.019	0.021	0.018	0.003
Random Forest	0.009	Mean	0.481	0.274	**0.349**	0.911
		Std	0.028	0.018	0.020	0.002
Gaussian Naive Bayes	0.003	Mean	0.207	**0.652**	0.314	0.753
		Std	0.014	0.050	0.011	0.023
Multi-layer Perceptron	0.462	Mean	0.658	0.202	0.306	**0.921**
		Std	0.056	0.036	0.040	0.002
SVC (*kernel=rbf*)	4.521	Mean	**0.750**	0.110	0.191	0.920
		Std	0.047	0.012	0.019	0.001
Dummy Classifier (*strategy=stratified*)	0.001	Mean	0.086	0.083	0.085	0.842
		Std	0.012	0.013	0.011	0.003
Dummy Classifier (*strategy=most freq.*)	0.001	Mean	0.0	0.0	0.0	0.913
		Std	0.0	0.0	0.0	0.0

From the analysis it immediately emerges that there is not a clear winner among the different classifiers. Each one has its strengths and weaknesses. The Extra Tree classifier, for instance, produces balanced precision and recall (around 33% and 34%) and the second best F1 score. Furthermore, it is among the fastest. Random Forest achieves the best F1 score of 35%, good precision (48%) and accuracy (91%) but below average recall. Naive Bayes has the highest recall of 65%, but a low precision of 21%. The opposite holds for the MLP, which has a high precision (66%) and the highest accuracy (together with SVC) of 92%, but a low recall (20%). The support vector classifier outperforms all other classifiers with respect to precision (75%) and accuracy, but it generates the lowest recall and is much slower compared to the other classifiers.

It can, therefore, be concluded that all five classifiers provide significantly better results than the baselines and therefore have the potential to be implemented in a real-world setting, depending on the desired metrics to optimize. To sum up the results, the following list shows the best classifier for each measure:

- Precision: Support Vector Classifier
- Recall: Gaussian Naive Bayes
- F1: Random Forest
- Accuracy: Multi-layer Perceptron

Since this machine learning application is built for the specific task of predicting booking events in the tourism domain, we focus on the peculiarity of this application. In general we can say that this is a low-risk task, since, wrongly predicting a positive instance (i.e., to consider as a booker a person who will not convert her request into a booking) is not so harmful and would only waste a few minutes to send an offer. Instead, missing a positive instance and ignoring a potential booker, without taking proper actions, would strongly hurt the hotel's revenue. Thus, it is reasonable to consider recall (i.e., the proportion of positive cases correctly retrieved) as the most important measure to optimize. Thus, naive bayes turns out to be the most promising classifier to be tested in a real-world scenario. Furthermore, in the next section we apply the subsampling technique to cope with the imbalanced class proportion in the original data and optimize for the recall metric.

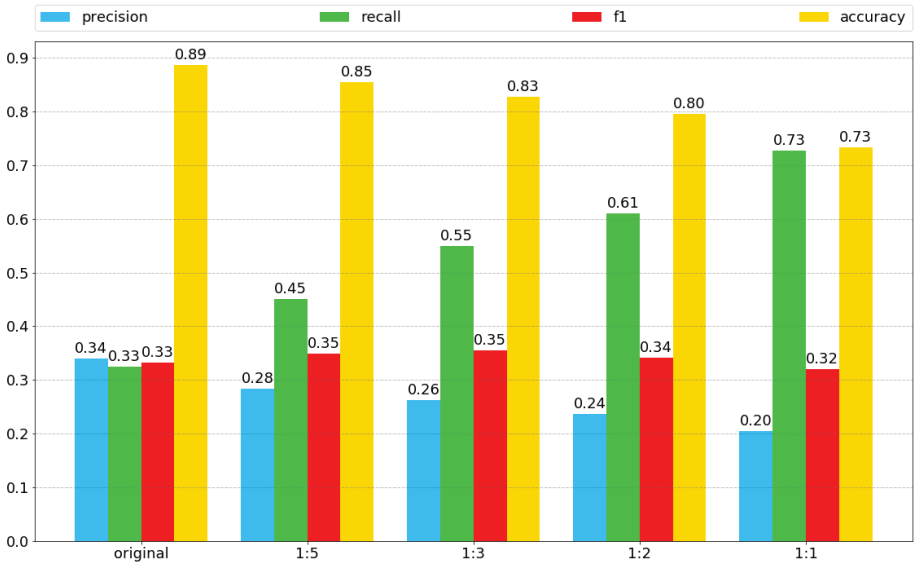
3.2. Class Imbalance Subsampling

We study the effect of the subsampling technique to cope with the class imbalance problem (Japkowicz & Stephen, 2002). In a real-world scenario the positive class (i.e., the class we want to predict) is mostly outnumbered by the negative class. In most cases it is therefore more relevant to correctly retrieve the positive instances rather than the negative ones. Nevertheless, due to the imbalanced number of examples belonging to the two classes, many classifiers will be biased towards the prediction of the majority class. To counter this problem, the proportion of positive and negative classes in the train set can be adjusted in order to remove this bias. As already mentioned, the analyzed dataset suffers strongly from class imbalance, i.e., only 8.62% of instances belong to the positive class (booking).

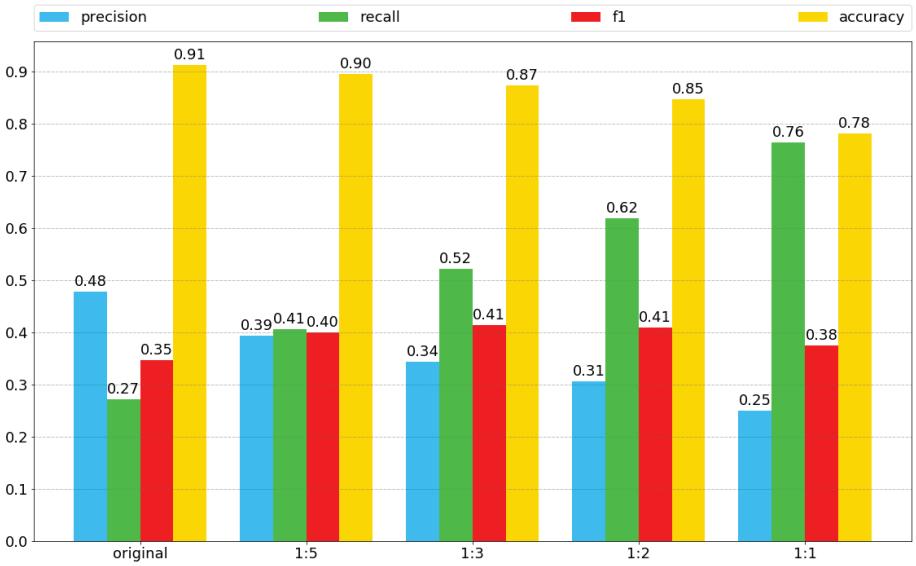
Hence, in order to improve the recall of the positive class we ran a set of experiments by enforcing different class proportions on the training set. Namely, the training set was modified in order to achieve a more balanced ratio of positive to negative cases, i.e., 1:5, 1:3, 1:2 and 1:1. For each ratio, the performances of the classifiers were tested using a 75/25 hold-out approach, repeated for 10 times (Tan, 2005). In each iteration the ratio of positive to negative class values of the 75% training set is modified, by randomly filtering out negative instances. The classifier is then trained on the modified set and finally tested on the remaining 25% test set with the original proportion of positive/negative observations. As for the previous experimental setup, averaging the results over 10 repetitions with random train/test splitting ensures more robust and significant measurements.

In Figure 2, we plot the results based on subsampling with different positive/negative ratios to alleviate the class imbalance problem. The results on the original dataset without subsampling are also reported for comparison. If we focus on precision and accuracy, it can be observed that all five classifiers performed best on the original dataset. By removing negative observations, the performances of precision and accuracy decrease, reaching the worst scores with the ratio of 1:1, i.e., the same amount of positive and negative observations. However, in terms of recall all classifiers perform worst on the originally balanced data and best with perfect class balance. The same trend can be observed for all five classifiers, i.e. removing negative entries increases recall as expected. When comparing methods for highest recall values (on the balanced proportions) across all five classifiers, we notice that naive bayes and the tree-based approaches achieve the lowest results, between 73% and 76%. Instead, MLP and SVC greatly improve their recall measure, compared to the original data distribution, scoring 83% and 87% respectively. Whereas the scores of precision decrease due to subsampling the original data, while just the gaussian naive bayes classifier performs best on the original proportion. For four out of five classifiers, the F1 score reaches the best performance with the proportion 1:3, i.e., the trade-off between precision and recall is the highest among the class ratios of positive and negative examples.

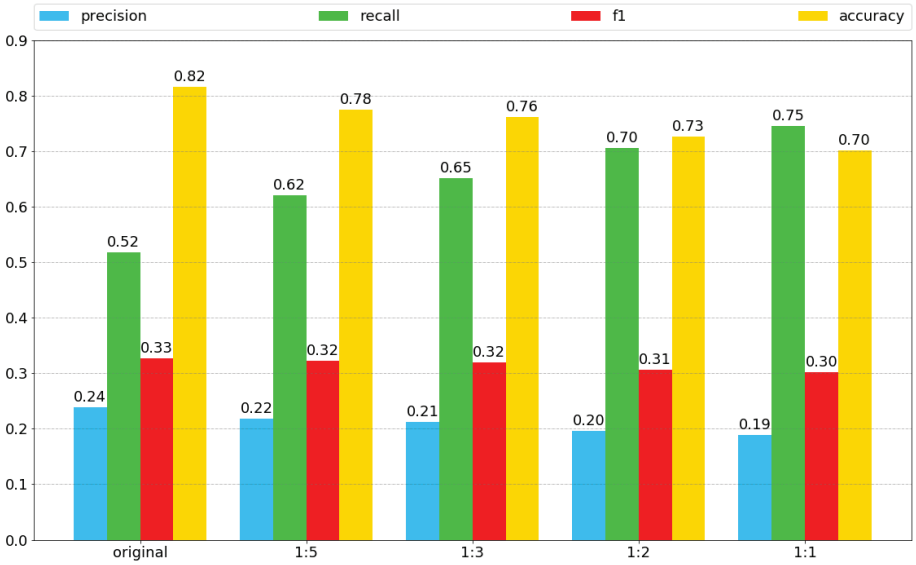
Figure 2: **Classification results for the five classifiers trained after subsampling negative class to achieve different positive/negative ratios (i.e., original, 1:5, 1:3, 1:2, 1:1)**



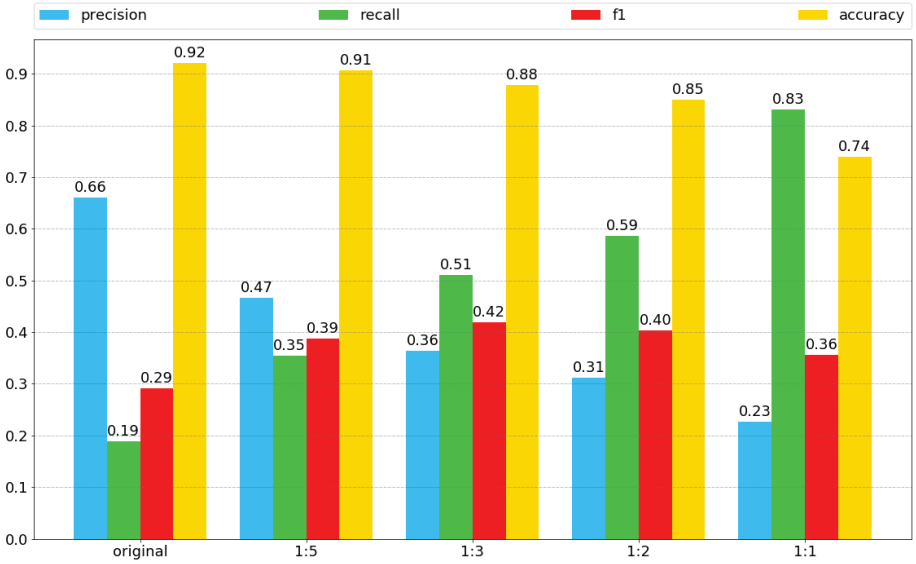
(a) Extra Tree



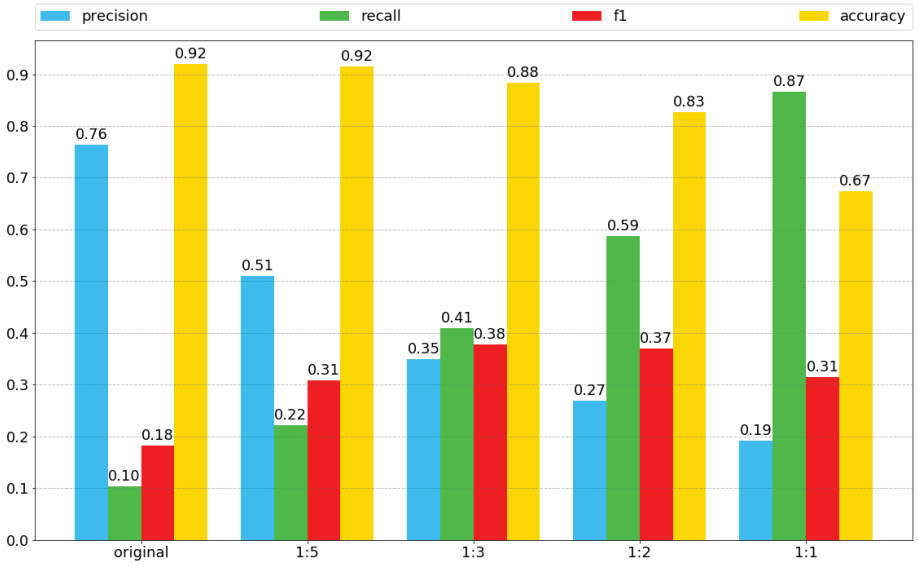
(b) Random Forest



(c) Gaussian Naive Bayes



(d) Multi-layer Perceptron



(e) Support Vector Classifier

To summarize, even though subsampling deteriorates precision and accuracy, it emerges that this technique, i.e., finding the best ratio of positive to negative instances, could help in improving the positive recall and getting a better recall/precision trade-off, while increasing the correctly predicted positive examples. The most balanced ratio (i.e., 1:1) produces the highest recall for each classifier, with the overall optimal achieved by MLP and SVC (i.e. above 82%). This combination of methods is recommended if we target to optimize recall, as discussed in Section 4.1. The ratio of 1:3 should, instead, be considered as the optimal proportion to train the classifiers in order to improve the recall/precision balance. In general, for random forest and MLP highest F1 scores were produced (around 42%).

The major business insight derived from this analysis is that in an ideal scenario, fitted to the collected results, we have approximately 10% of the incoming requests turning into a booking and we are able to apply a classifier that achieves a recall of 80% and a precision of 20%. This means that, by using our approach as a pre-filter and by prioritizing 40% of the incoming requests (automatically selected by the classifier), the receptionist would correctly serve 80% of all actual bookers, saving, at the same time, 60% of his working time. This intuition is schematized in the graph in Figure 3.

Figure 3: **Comparison of the theoretic workload and booking success considering all the incoming requests and the requests pre-filtered by our automated approach**

3.3. Feature Selection

Feature selection is a technique that reduces the number of features in the dataset with the aim of improving the classification performance by counter- acting overfitting (Liu & Motoda, 2007). Two main categories of feature selection exist: filter approach and wrapper approaches. In the filter approach, the redundant/irrelevant features are removed before applying the classification algorithm (Guyon & Elisseeff, 2003). Features are filtered, regardless of the classifier, using score functions such as chi-squared, mutual information (entropy), or Gini index. Wrapper methods, instead, select the best combination of features by comparing the performances of different feature subsets for a specific classification model (Kohavi & John, 1997).

In our work, we focused on the filter approach applied on the five selected classifiers. We evaluated the classification performances over the *i* best features with i = *1, 2, . . ., n*, where *n* is the total number of features in our dataset. Furthermore, we discuss the feature ranking that emerges from the filter method in order to provide more insights on the data. The experiments in this section were performed using a standard time-based 75/25 hold-out split, i.e., 75% of the older data were used as a training set to create a ranking over the features and learn the model on the selected subset of features, while the remaining 25% of the more recent data were used to test the trained classifier on unseen data. This evaluation protocol was required to speed-up the high computational time requested for replicating the experiments for all the feature sets. We decided to always use the same portion of the dataset to learn the feature ranking and train each model. This technique ensures to be consistent across different models and to generate a unique set of features at each step. Furthermore, since the feature selection is performed by an external measure (i.e., independent from the classifiers), the feature ranking is fixed across all the classification algorithms.

In Table 5, we report the ranking of the features based on the entropy score computed on 75% of the less recent data. This feature importance score is derived from the decision tree representation of the classification task. The score is calculated as the decrease in node entropy after the splitting, weighted by the probability of reaching that node. The score is then averaged for each feature and normalized such that the scores for all the features sum up to 1. It can be seen that the feature "CR_RequestedDaysBeforeArrival" clearly outscores the other features, with an importance score of 0.31. The feature "CG_CountryCode" also gets a reasonably high score of 0.2. Furthermore, both "CR_Duration" and "CR_SourceOfBusiness" still perform reasonably with respect to the less important features (around 0.12 of entropy-based score). The features from position 5 to 16, in contrast, score poorly in terms of information gain (below 0.1), that means they provide a relatively small contribution to discriminate between the classes.

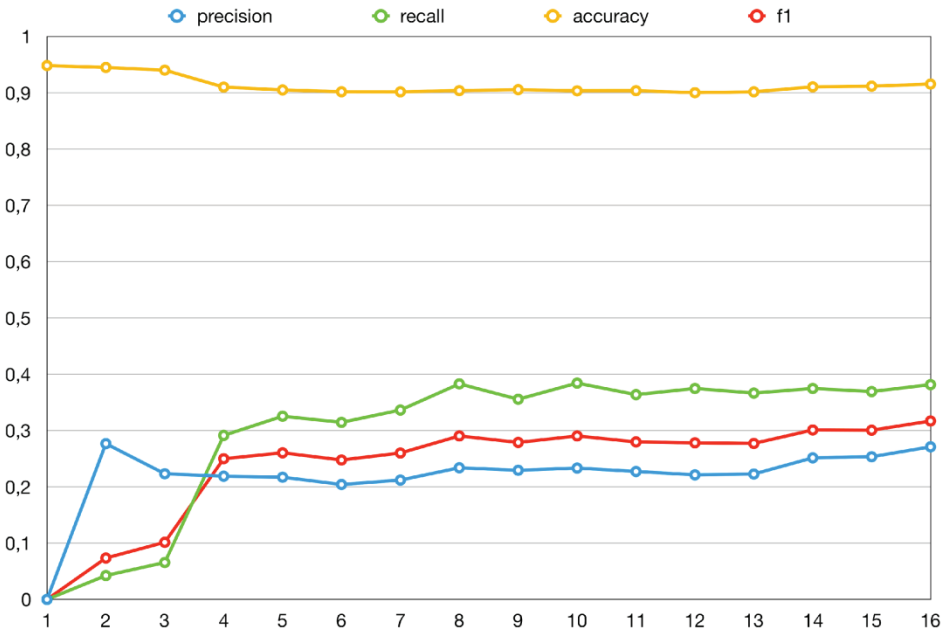
Table 5: **Feature ranking for each classifier on entropy-based score**

Rank	Feature	Score
1	CR_RequestedDaysBeforeArrival	0.312
2	CG_CountryCode	0.202
3	CR_SourceOfBusiness	0.120
4	CR_Duration	0.116
5	CR_Adults	0.062
6	CG_Gender	0.046
7	CR_Children	0.024
8	CG_LanguageId	0.024
9	L_SendNachhakEmail	0.022
10	CR_Season	0.019
11	CR_Age_4-10	0.018
12	CR_Age_0-3	0.010
13	CR_Age_11-17	0.010
14	L_RecallEmailSended	0.008
15	L_UseWebTemplate	0.006
16	CWM_Message	0.000

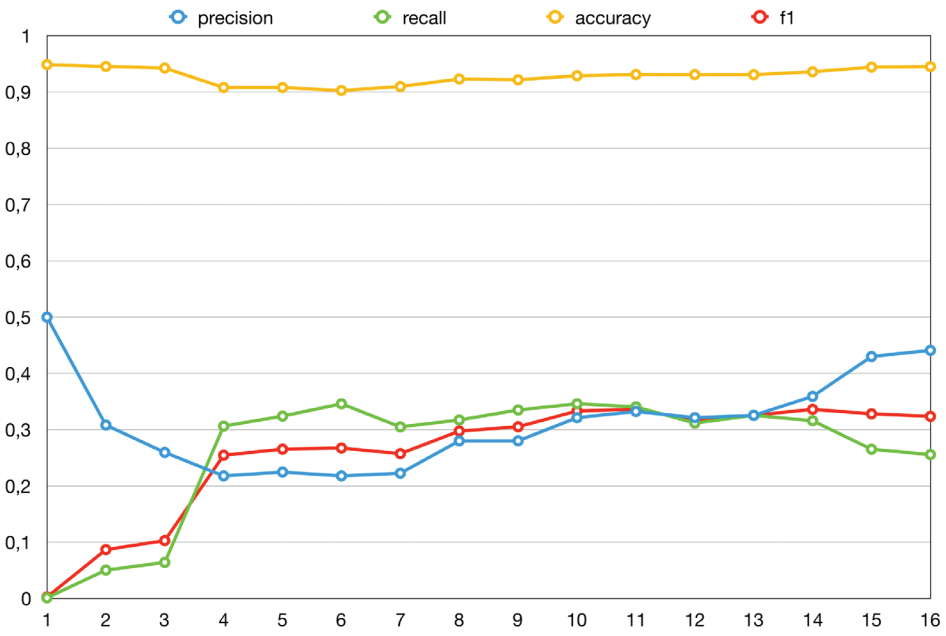
In Figure 4, we show the graphs representing the classification performances of the five selected classifiers trained on the best *i* features (with *i = 1, 2, . . ., n*, displayed on the x-axis), derived from the entropy-based ranking. The results are produced in a single run of the time-based hold-out evaluation protocol.

It is interesting to notice the different behaviors of the classifiers that emerge from the graphs with respect to the feature selection. Decision tree classifiers (i.e., random forest and extra tree) are less affected by the selection of the features, as depicted in Figures 4a and 4b: from 4 up to 16 features they reach a stable and optimal performance on the metrics related to the positive class. The only exception is that random forest gets the absolute best precision with just one feature. This is due to the fact that very few positive instances are predicted by the classifier, because of the strong class imbalance, and thus it is much easier to get a higher precision. A similar pattern is observed for the support vector classifier in Figure 4e. In this case a significant boost in precision (around 30%) occurs when the 8 best features are selected, and afterwards it stays stable up to the complete set of features. The two other classifiers instead similarly show that their performances are much more affected by feature selection. For both multi-layer perceptron and naive bayes (shown in Figures 4c and 4d), the best results of F1 measure are achieved between 8 and 14 features. MLP presents a more stable behavior across the optimal range of features, while NB behaves in a more idiosyncratic way, with a peak in precision for 9 features and a peak in recall for 13 features.

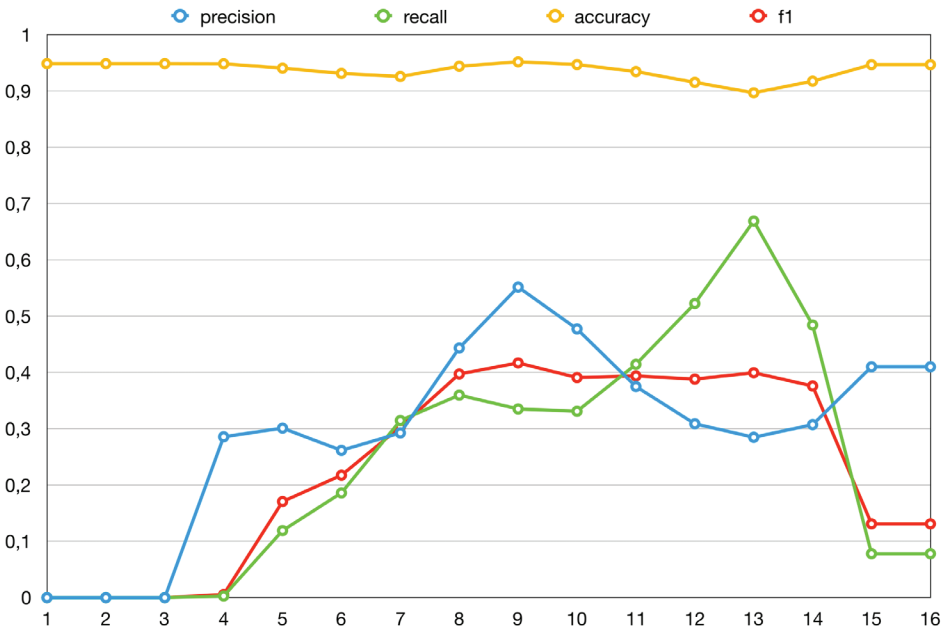
Figure 4: **Classification results for the five classifiers trained on the best *i* features as ranked by the entropy-based feature selection. On the x-axis the number *i* of best features considered, on the y-axis the performance values.**



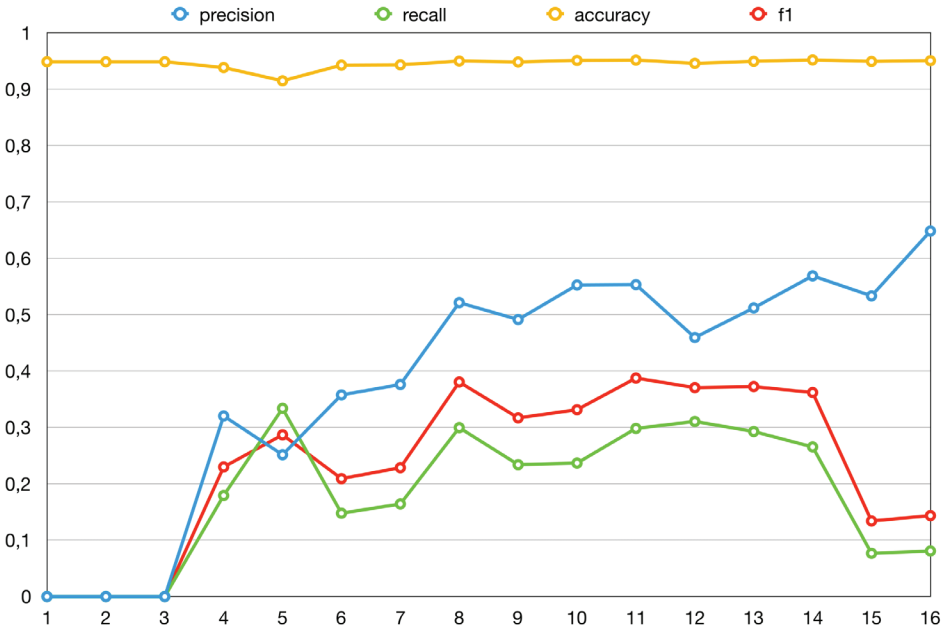
(a) Extra Tree



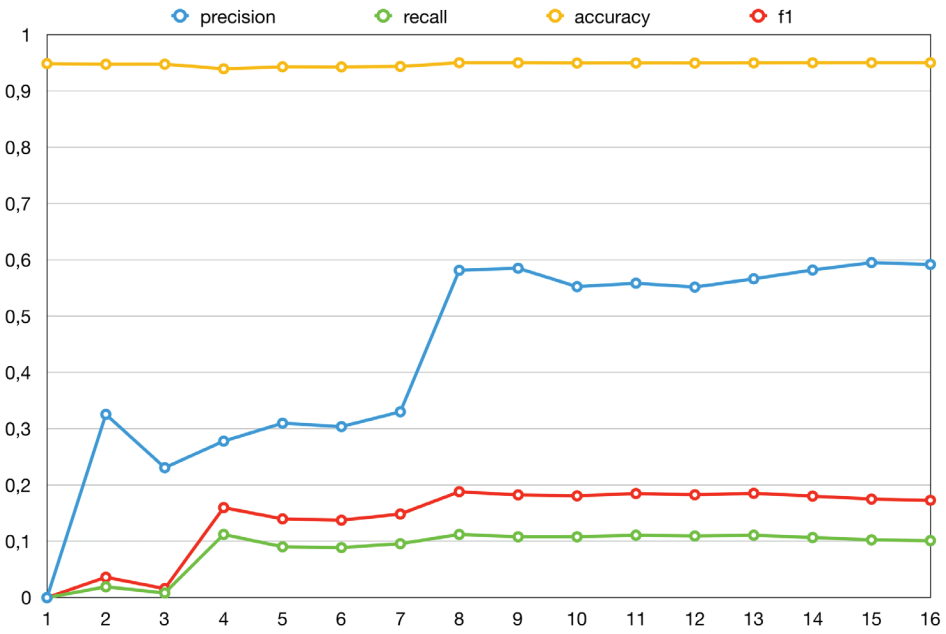
(b) Random Forest



(c) Gaussian Naive Bayes



(d) Multi-layer Perceptron



(e) Support Vector Classifier

3.4. Feature Value Analysis

We qualitatively explore the semantics behind the features selected as important in the feature selection phase. In particular, we aim at enhancing the comprehension of the booking process and increasing the business value of the analysis. Thus, we inspect differences between feature values of observations belonging to the two class bins (i.e., "Book" and "NotBook").

In Table 6, we report the average value of the numeric features for the positive and the negative observations. We notice that the average of some of the values between the positive and the negative records differ. The average number of adults "CR_Adults" is, for instance, slightly lower for the booked requests than for the non-booked requests. Similarly, the number of children and all three age spans ("CR_Children", "CR_Age_0-3", "CR_Age_4-10", "CR_Age_11-17") show a slightly lower average for booked requests. This highlights the general tendency of larger groups of people (or bigger families) to be more exploratory in their requests, showing less propensity to actual booking than smaller ones.Furthermore, the average duration of the stay ("CR_Duration") of the booked requests is more than half a day shorter than the average duration of the non-booked requests, corroborating the intuition that requests for longer (thus more expensive) stays are in general less probable to be converted into bookings. The most significant difference in the table can be observed on the average number of days between request and arrival date ("CR_RequestedDaysBeforeArrival"). In fact, we observe that the positive requests were sent on average more than 8 days later than the negative ones, thus the closer the request date to the trip date the more likely the conversion.

Table 6: **Average value of the numeric features conditioned on the class**

Feature	Measure	NotBook	Book
CR_Adults	Frequency	2.475	2.442
CR_Children	Frequency	0.551	0.327
CR_Age_0-3	Frequency	0.125	0.073
CR_Age_4-10	Frequency	0.307	0.199
CR_Age_11-17	Frequency	0.119	0.055
CR_Duration	Days	5.779	5.043
CR_RequestedDaysBeforeArrival	Days	57.760	49.276
CWM_Message	Words	1.170	2.682

In Table 7, we report significant class frequency counts for categorical features, i.e., we compute the percentage of bookings given some specific feature values. It becomes evident that significant differences exist in the booking probability for some of these categorical values. For instance, if we consider the feature "CG_CountryCode" we notice a different behavior with respect to the measured booking rate. The country code 65 (unknown) corresponds to a booking-rate of just 1%. This implies that a request in which the country code is not specified by the user is very likely to not be converted into a booking. Vice versa, specific codes largely increase the probability of conversion. For the country codes 17 (Germany) and 34 (Italy) a booking-rate

of 18% was registered in the database. The increase is even higher for country code 28 (Croatia), with a 30% booking-rate, but with less observations than the other mentioned countries. Instead, the attribute "CG_Gender" slightly impacts the booking probability. In particular, the booking-rate conditioned on each gender type (namely, 0: male, 1: female, 2: company, 3: family) is close to the baseline of 8%. Similar conclusions can be drawn by observing the "CR Season" attribute, whereas the booking-rate is slightly higher for season 0 (winter) at around 10%. Furthermore, we can see that the language ("CG_LanguageId") selected for the communication with the hotel has a larger impact on the booking behavior. Indeed, German-speaking (id: 1) and Italian-speaking (id: 2) persons have a booking-rate of 7% and 9% respectively (in line with the average of the dataset), whereas English-speaking (id: 3) persons 21%. This analysis might prove that booking requests coming from outside the surrounding area of the region (i.e., not from German or Italian speakers) have a much higher chance to turn into an actual reservation. Looking at the booking-rate of the feature "CR_SourceOfBusiness" we see the most diverse results. Specifically, the largest difference can be noticed between categories 6 and 20. Indeed, category 6 has a booking-rate of 91% while category 20 a booking-rate of just 1%. The former is mapped to a source of request that is manually used by receptionists; thus, it might represent requests coming in via phone call. The latter is mapped to a request portal related to a specific skiing area. Finally, the last three listed features are related to specific parameters of the system. In particular, when a recall e-mail was sent ("L_RecallEmailSended" = 1) the booking-rate is lower. The same applies when the customer is allowed to send reminder emails ("L_SendNachhakEmail" = 1). We can also see that the booking rate for traditional e-mail offers ("L_UseWebTamplate" = 0) is higher than for offers in the form of a website ("L_UseWebTamplate" = 1). From these observations it could be inferred that more traditional and personal communication channels result in a higher likelihood of an actual booking.

Table 7: **Occurrences of different categorical variable values in the two classes. For each feature we report the most interesting values, i.e., when a significant difference exists in the booking-rate conditioned on different values of the feature**

Feature	Value	NotBook	Book	Booking-rate
CG_CountryCode	2	481	162	25 %
CG_CountryCode	12	1047	188	15 %
CG_CountryCode	17	6102	1330	18 %
CG_CountryCode	28	224	94	30 %
CG_CountryCode	34	10901	2385	18 %
CG_CountryCode	65	32422	380	1 %
CG_Gender	0	28222	3162	10 %
CG_Gender	1	21369	1644	7 %
CG_Gender	2	1580	101	6 %
CG_Gender	3	964	12	1 %
CG_LanguageId	1	24694	1973	7 %
CG_LanguageId	2	26147	2594	9 %
CG_LanguageId	3	1294	352	21 %
CR_Season	0	27686	2973	10 %
CR_Season	1	24449	1946	7 %
CR_SourceOfBusiness	0	20683	2365	10 %
CR_SourceOfBusiness	1	5401	774	13 %
CR_SourceOfBusiness	2	22486	1121	5 %
CR_SourceOfBusiness	3	1376	128	9 %
CR_SourceOfBusiness	4	1150	174	13 %
CR_SourceOfBusiness	6	29	302	91 %
CR_SourceOfBusiness	20	452	3	1 %
L_RecallEmailSended	0	43259	4571	10 %
L_RecallEmailSended	1	8876	348	4 %
L_SendNachhakEmail	0	8076	1820	18 %
L_SendNachhakEmail	1	44059	3099	7 %
L_UseWebTemplate	0	38750	4249	10 %
L_UseWebTemplate	1	13385	670	5 %

DISCUSSION AND CONCLUSION

We described a real-world application of machine learning techniques and data analysis in the context of e-tourism. To the best of our knowledge, this is a first attempt to leverage automated data mining to predict the likelihood of an incoming request of quotation converting into a booking. Nevertheless, our work was inspired and grounded on previous research on booking cancellation prediction (Antonio et al., 2017) and tourism demand forecasting (Pan et al., 2012).

Namely, we conducted an extensive experimental study on a dataset constructed from requests for quotation collected by a 4-star hotel in South Tyrol (Italy) during the period 2014-2019. The task was to predict whether a request will convert into a booking or not, given a set of 16 engineered and semantically meaningful features. A large exploratory study with 16 different supervised classification models is conducted to select the 5 best performing ones according to different classification metrics, to be further exploited for more insightful analysis. The best performing models in this analysis achieved convincing results in the booking prediction task (RQ 1). Specifically, Multi-layer Perceptron and Random Forest resulted as the most promising models overall, registering the highest F1 score (around 42%), derived from a high recall (between 50% and 60%), against a reasonable precision (above 30%). Those results clearly outperformed benchmark techniques, and confirmed the findings by Antonio et al., (2017) of tree-based and neural network algorithms being the most suitable for tourist behavior prediction.

We further optimize the classification performance towards the important business metric of recall (i.e., percentage of booking events correctly retrieved), by running a set of experiments centered on majority class subsampling, to cope with class imbalance (RQ 2). Indeed, as discussed in Section 3.2, the most important insight derived from this analysis is that a MLP classifier trained on a equally re-balanced dataset is able to optimally support the receptionist work by prioritizing 80% of the correctly identified bookers, saving 60% of her working time. This trend is mildly corroborated in different case studies in the related field of tourism demand and revenues forecasting. Indeed, MLP-based forecasting algorithms produced convincing results (Claveria & Torra, 2014, Lado-Sestayo & Vivel-Búa, 2019). On the other hand, Höpken et al., (2017) proved the capability of a simpler nearest neighbor approach to outperform standard statistical approaches, like linear regression.

Finally, a thorough analysis on feature selection and feature importance is performed. This analysis was intended with the dual goal of optimizing classification performance (RQ 3) and providing more insights on the features' semantics and their impact on the prediction task (RQ 4). In fact, time-related features of the request, such as the number of days prior of the stay and the duration of the stay, as well as qualitative features, such as the country of the inquirer and the business source of the request, turned out to be the most important attributes to discriminate a booking request. Those findings were partially confirmed in previous experimental research. For instance, Antonio et al., (2017) reported the country of the customer as one of the most relevant independent variables to predict booking cancellation. Instead, in contrast with our results, the duration of the stay and the booking time showed a marginal impact on the conversion or cancelation rate (Antonio et al., 2017, Cezar & Ogüt, 2016). In conclusion, promising results emerged from the exploratory analysis of an automated approach, in the e-tourism domain, used as a filtering support for requests for quotation which will most probably convert into a booking. Hence, this work could be easily extended and replicated by practitioners and hotel managers, applying standard machine learning techniques to the real-world data of an accommodation facility, hence, improving the capability of the managing software of prioritizing more profitable incoming requests, and, finally, increasing revenues.

The main limitation of the presented research concerns the limited amount of data related to a single hotel and the scarce set of (non-personalized) features involved in the study. Thus, future directions of this research should concern enriching the dataset, including more data from different hotels, as well as additional features, such as customers' demographics, room characteristics, or features describing the request and booking process. For instance, it would be interesting to process the communication (i.e., e-mails and chats) between the accommodation and the customer with natural language processing techniques in order to extrapolate text-related features. An additional direction for the work is the one of exploiting more sophisticated techniques to infer in a systematic way the impact of the feature and their importance in the prediction task.

REFERENCES

Adil M., Ansari M. F., Alahmadi A., Wu, J.-Z., & Chakrabortty, R. K. (2021). Solving the Problem of Class Imbalance in the Prediction of Hotel Cancellations: A Hybridized Machine Learning Approach. *Processes*, 9(10). https://doi.org/10.3390/pr9101713

Ali, A., Shamsuddin, S. M., & Ralescu, A. L. (2013). Classification with class imbalance problem: a review. *Int. J. Advance Soft Compu. Appl*, 5(3).

Antonio, N., De Almeida, A., & Nunes, L. (2017). Predicting hotel booking cancellation to decrease uncertainty and increase revenue. *Tourism and Management Studies*, 13(2), 25–39. https://doi.org/10.18089/tms.2017.13203

Assaf, A. G., Li, G., Song, H. &Tsionas, M. G. (2019). Modeling and forecasting regional tourism demand using the bayesian global vector autoregressive (BGVAR) model. *Journal of Travel Research*, 58(3), 383–397. https://doi.org/10.1177/0047287518759226

Cezar, A., & Ogüt, H. (2016). Analyzing conversion rates in online hotel booking: The role of customer reviews, recommendations and rank order in search listings. *International Journal of Contemporary Hospitality Management*, 28(2), 286–304. https://doi.org/10.1108/IJCHM-05-2014-0249

Chen, K.-Y., & Wang, C.-H. (2007). Support vector regression with genetic algorithms in forecasting tourism demand. *Tourism Management*, 28(1), 215– 226. https://doi.org/10.1016/j.tourman.2005.12.018

Chiang, W.-C., Jason, C. H. Chen, & Xu, X. (2007). An overview of research on revenue management: Current issues and future research. *International Journal of Revenue Management*, 1(1), 97-128. https://doi.org/10.1504/IJRM.2007.011196

Çiftçi, S. F., & Çizel, B. (2020). Predictors of e-trust for web-based travel intermediaries: a survey on Istanbul visitors. *Journal of Hospitality and Tourism Technology* 11(4), 667–680. https://doi.org/10.1108/JHTT-02-2019-0037

Claveria, O., & Torra, S. (2014). Forecasting tourism demand to Catalonia: Neural networks vs. time series models.*Economic Modelling*, 36, 220–228. https://doi.org/10.1016/j.econmod.2013.09.024
Egger, R. (2022). Machine Learning in Tourism: A Brief Overview. In Egger, R. (Eds.) *Applied Data Science in Tourism*: Interdisciplinary Approaches, Methodologies, and Applications (pp.85-107), Springer, Cham. https://doi.org/10.1007/978-3-030-88389-8_6
Fiori, A. M., & Foroni, I. (2020). Prediction accuracy for reservation-based forecasting methods applied in revenue management. *International Journal of Hospitality Management,* 84. https://doi.org/10.1016/j.ijhm.2019.102332
Gretzel, U., Sigala, M., Xiang, Z., & Coo, C. (2015). Smart tourism: foundations and developments. *Electronic Markets*25, 179–188.https://doi.org/10.1007/s12525-015-0196-8
Gurney, K. (1997). *An Introduction to Neural Networks*, USA: Taylor & Francis.
Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3, 1157–1182.
Höpken, W., Ernesti, D., Fuchs, M., Kronenberg, K., & Lexhagen, M. (2017). Big data as input for predicting tourist arrivals. In Schegg, R., & Stangl, B. (Eds.), *Information and communication technologies in tourism 2017* (pp. 187–199), Springer. https://doi.org/10.1007/978-3-319-51168-9_14
Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent data analysis,* 6(5), 429–449. https://doi.org/10.3233/IDA-2002-6504
Kimes, S. E., & Wirtz, J. (2003). Has revenue management become acceptable?: Findings from an international study on the perceived fairness of rate fences. *Journal of Service Research,* 6(2), 125–135. https://doi.org/10.1177/1094670503257038
Khatibi, A., Belém, F., Couto da Silva, A. P., Almeida, J. M., & Gonçalves, M. A. (2020). Fine-grained tourism prediction: Impact of social and environmental features. *Information Processing & Management*, *57*(2), 102057. https://doi.org/10.1016/j.ipm.2019.102057
Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence,* 97(1), 273 – 324. https://doi.org/10.1016/S0004-3702(97)00043-X
Lado-Sestayo, R., & Vivel-Búa, M. (2018). Profitability in the hotel sector: a PLS approach. *Journal of Hospitality and Tourism Technology*, 9(3), 455–470. https://doi.org/10.1108/JHTT-10-2017-0118
Lado-Sestayo, R., & Vivel-Búa, M. (2019). Hotel profitability: a multilayer neural network approach. *Journal of Hospitality and Tourism Technology*, 11(1), 35–48. https://doi.org/10.1108/JHTT-08-2017-0072
Li, X., Pan, B., Law, R., & Huang, X. (2017). Forecasting tourism demand with composite search index. *Tourism management,* 59, 57–66. https://doi.org/10.1016/j.tourman.2016.07.005
Liu, H., & Motoda, H. (2007). *Computational Methods of Feature Selection (Chapman & Hall/CRC Data Mining and Knowledge Discovery Series),* Chapman & Hall/CRC.
Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*, Cambridge: The MIT Press.
Pan, B., Chenguang Wu, D., & Song, H. (2012). Forecasting hotel room demand using search engine data. *Journal of Hospitality and Tourism Technology*, 3(3), 196-210. https://doi.org/10.1108/17579881211264486
Pan, B., & Yang, Y. (2017). Forecasting destination weekly hotel occupancy with big data. *Journal of Travel Research*, 56(7), 957–970. https://doi.org/10.1177/0047287516669050
Rokach, L., & Maimon, O. Z. (2014). Data mining with decision trees: theory and applications. World Scientific Publishing Co., Inc., River Edge. https://doi.org/10.1142/9097
Schölkopf, B., & Smola, A. J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. The MIT Press. https://doi.org/10.7551/mitpress/4175.001.0001
Schwartz, Z., Uysal, M., Webb, T., & Altın, M. (2016). Hotel daily occupancy forecasting with competitive sets: a recursive algorithm. *International Journal of Contemporary Hospitality Management*, 28(2), 267–285. https://doi.org/10.1108/IJCHM-10-2014-0507
Tan, P.-N. (2005). *Introduction to Data Mining*, Pearson.
Thomas, E., Ferrer, A. G., Lardeux, B., Boudia, M., Haas-Frangii, C. & Agost, R. A. (2019). Cascaded machine learning model for efficient hotel recommendations from air travel bookings. *Proceedings of the 12th ACM Conference on Recommender Systems, ACM RecSys Workshop on Recommenders in Tourism* (9-16), Copenhagen, Denmark. https://doi.org/10.1145/3240323.3240341
Thabtah, F., Hammoud, S., Kamalov, F., & Gonsalves, A. (2020). Data imbalance in classification: Experimental evaluation. *Information Sciences*, *513*, 429-441. https://doi.org/10.1016/j.ins.2019.11.004
Webb, T., Schwartz, Z., Xiang, Z., & Singal, M. (2020). Revenue management forecasting: the resiliency of advanced booking methods given dynamic booking windows. *International Journal of Hospitality Management*, 89. https://doi.org/10.1016/j.ijhm.2020.102590
Wong, E., Rasoolimanesh, S. M., & Sharif, S. (2020). Using online travel agent platforms to determine factors influencing hotel guest satisfaction. *Journal of Hospitality and Tourism Technology*, 11, 425–445. https://doi.org/10.1108/JHTT-07-2019-0099
Xie, K. L., & Lee, Y. J. (2020). Hotels at fingertips: informational cues in consumer conversion from search, click-through, to book. *Journal of Hospitality and Tourism Technology*, 11(1), 49–67. https://doi.org/10.1108/JHTT-03-2017-0026
Yang, X., Pan, B., Evans, J. A., & Lv, B. (2015). Forecasting Chinese tourist volume with search engine data. *Tourism Management,* 46, 386 – 397. https://doi.org/10.1016/j.tourman.2014.07.019

Please cite this article as:
Runggaldier, S., Sottocornola, G., Janes, A., Stella, F., & Zanker, M. (2023). Booker Prediction from Requests for Quotation via Machine Learning Techniques. *Tourism and Hospitality Management, 29(1)*, 25-43. https://doi.org/10.20867/thm.29.1.3

Creative Commons Attribution – Non Commercial – Share Alike 4.0 International

APPENDIX

A. MACHINE LEARNING BACKGROUND

A.1. Classification algorithms

Classification (more formally, supervised classification) is the specific area of machine learning that aims to assign objects to one of several predefined categories.

The objects, that represent the input of a classification task, are called records, instances or observations. Each of these records

is characterized by a tuple (X, y), where X represents the set of attributes or features of the object and y its class label. The attributes can be both numerical (i.e. continuous values) or categorical (i.e. discrete, cardinal values), while the class label must be categorical. Classification is the task of learning a target function f(X) -> y that maps each attribute set X to one of the class labels y.

In the following, we describe the most common classification algorithms - i.e., the procedures responsible to learn from the data the mapping function f(X) -> y.

Decision tree classifier - Decision tree classification models are a family of predictive models used for supervised classification, represented by a tree structure composed by branches and leaves. A decision tree is used as a predictive model to go from observations of the features X of an instance (associated with the branches) to conclusions about the instance's target class y (associated with the leaves).

A decision tree is learned by splitting the data into subsets, based on attribute value tests (i.e., logic statements on attribute values). This process is repeated on each derived subset in a recursive manner, called recursive partitioning. The recursion is completed when the whole subset belongs to the same target class, or when splitting no longer adds value to the classification performance. The quality of the split is determined based on homogeneity metrics of the target class within the subsets (e.g., through Gini impurity or information gain).

A peculiar type of classification procedure that enjoys a lot of success in the machine learning community is represented by ensemble learning. The general idea is to boost classification performances by averaging across the predictions of an ensemble of classifiers instead of relying on a single one. In the context of decision trees **Random forest** and **Extra tree** classifiers were introduced.

Random forest is an ensemble learning method that operates by constructing a multitude of randomized and uncorrelated decision trees at training time and outputting the class that is the mode of the classes of the individual trees. The main advantage of random forests over single decision trees is the capability of avoiding overfitting to the training set, thanks to the bagging (or bootstrap aggregating) procedure. Extra tree (referred to as extreme random forest) is a slight variation of the random forest classifier, in which also the attribute split selection is randomized. This leads to more diversified trees and less splits to be evaluated during training, speeding up the tree building procedure.

Naive Bayes classifier - Naive Bayes classifier is a probabilistic classification algorithm, based on the Bayes theorem and on the strong (naive) assumption of conditional independence between features in X given the class y.

In this setting we consider the classification problem from a probabilistic perspective. The class variable is assumed to have a non-deterministic relationship with the attributes. Thus, we treat the set of attributes as a set of random variables X and the class as random variable y and capture their relationship stochastically with the conditional probability P(y|X). This conditional probability is also known as the posterior probability of y, as opposed to its prior probability, P(y).

Accurately estimating the posterior probabilities for every possible combination of class label and attribute values is a difficult problem because it requires a very large training set, even for a moderate number of attributes. The well-known Bayes theorem is leveraged in order to express the posterior probability in terms of the prior probability P(y), the class-conditional probability (or likelihood) P(X|y), and the prior probability P(X), as: $P(y|X) = P(X|y)\,P(y) / P(X)$.

During the training phase, the algorithm learns the likelihood conditional probability P(X|y) and the prior probabilities P(X) and P(y) based on information gathered from the training data. In particular, to estimate the class-conditional probability P(X|y) the naive assumption comes in handy. Specifically, a naive Bayes classifier assumes that the attributes, i.e., the components of X,are conditionally independent given the class label y. Thus the class-conditional probability can be estimated for each attribute X_i in the attribute set $X = \{X_1, X_2, \ldots, X_d\}$ independently, just conditioned on the class y, instead of conditioning on every possible combination in X.

Finally, different assumptions can be drawn on the nature of the class-conditional distributions, corresponding to variants of the naive bayes classifier. Whereas multinomial and Bernoulli naive Bayes classifiers are used for discrete/categorical attributes, gaussian naive Bayes is used to deal with continuous values.

Artificial neural network - The study on artificial neural network (ANN) was inspired by the attempt to simulate biological neural systems. Analogous to human brain structure, an ANN is composed of an inter-connected assembly of nodes and directed links (edges) between the nodes.

The simplest ANN model is called **Perceptron**. The perceptron consists of two types of nodes: input nodes, which are associated with the input features X, and an output node, which represent the model output y (i.e., the target class). The nodes in a neural network's architecture are commonly known as neurons. In a perceptron, each input node is connected via a weighted link to the output node. The weighted link emulates the strength of synaptic connections between neurons. As in biological neural systems, training a perceptron model amounts to adapting the weights of the links until they fit the input-output relationships of the underlying data.

A perceptron computes its output value y', by performing a weighted sum on its inputs X_i in X, subtracting a bias factor b from the sum, and then applying a non-linear transformation g(.), called activation function (e.g., a sigmoid function). The weights W_i associated with each neuron X_i, as well as the bias factor b, are learned by the algorithm via gradient descent minimization of the loss function (e.g., cross-entropy) computed between predicted output y' and actual target class y.

The most common ANN supervised classifier is the **Multi-layer perceptron** (MLP) or **Feedforward neural network** (FNN), that introduces a more complex structure to the simple perceptron described so far. The network may contain several intermediary layers between its input and output layers. Such intermediary layers are called hidden layers and the nodes embedded in these layers are called hidden neurons. This allows FNN to model more complex relationships between the input and output variables. The process of learning the (hidden) weights of a MLP from data is a non-trivial task. Thus, an efficient algorithm based on gradient descent, called Backpropagation has been developed.

A.2. Evaluation metrics

The evaluation of the goodness of a classification model is a crucial aspect in machine learning. Once a model is trained, the learned function is tested on a separated set of homogeneous labeled data (i.e., *test set*). The class predicted for each observation is compared with the actual class and the *confusion matrix* for the classification task is built.

In the case of a binary classification (i.e., with positive and negative classes), the confusion matrix is a simple 2x2 table that counts the occurrences for each predicted/real class combination. It reports the number of *true positives* (TP, i.e., the number of instances that belong to the positive class and which have been predicted as such), *false positive* (FP, i.e., the number of instances that belong to the negative class wrongly predicted as positive), *true negatives* (TN, i.e., the number of instances that belong to the negative class and which have been predicted as such), and *false negatives* (FN, i.e., the number of instances that belong to the positive class wrongly predicted as negative), collected on the test data.

Directly from the confusion matrix, we are able to introduce the following metrics, that measure the classification performance:

- *Precision* = TP / (TP + FP)
- *Recall* = TP / (TP + FN)
- *F1-score* = 2TP / (2TP + FP + FN)
- *Accuracy* = TP + TN / (TP + FP + TN + FN)

Precision and recall highlight different aspects of the classifier performance. Precision shows the proportion of predicted positive examples that are actually positive, while recall indicates the proportion of positive instances that are successfully retrieved by the classifier. F1-score is the harmonic mean of precision and recall and represents a good summary of the two metrics. Finally, accuracy is used to assess the classification performance independently from the class: it indicates the share of correctly predicted examples (both in positive and negative classes). This is generally used as a standard metric for classification tasks, but in some applications, i.e., when there is a strong imbalance between the classes, its value is strongly dependent on the proportion of the majority (negative) class. Thus, it is preferable to look at other class-dependent metrics, such as precision and recall.

B. COMPLETE RESULTS

Table B1: **Mean scores of the 10-fold cross-validation of all classifiers**

Classifier	Precision	Recall	F1	Accuracy
Extra Tree	0.344	0.330	0.336	0.888
Random Forest	0.482	0.275	0.350	0.912
Gaussian Naive Bayes	0.208	0.653	0.314	0.754
Multi-layer Perceptron	0.658	0.203	0.307	0.922
SVC (kernel=rbf)	0.750	0.110	0.192	0.920
Logistic Regression	0.448	0.037	0.068	0.913
Stochastic Gradient Descent Classifier	0.279	0.255	0.233	0.868
Perceptron	0.293	0.227	0.194	0.869
Passive Aggressive	0.291	0.176	0.117	0.835
Decision Tree	0.334	0.352	0.343	0.884
K-nearest Neighbors	0.464	0.188	0.267	0.911
Radius Neighbors	0.451	0.136	0.208	0.911
Linear Discriminant Analysis	0.437	0.037	0.067	0.913
Quadratic Discriminant Analysis	0.200	0.729	0.312	0.721

Classifier	Precision	Recall	F1	Accuracy
SVC (kernel=sigmoid)	0.163	0.157	0.158	0.857
SVC (kernel=linear)	0.000	0.000	0.000	0.914
Dummy Classifier (strategy=stratified)	0.086	0.084	0.085	0.842
Dummy Classifier (strategy=most frequent)	0.000	0.000	0.000	0.914

Table B2: **Standard deviation of the scores of the 10-fold cross-validation of all classifiers**

Classifier	Precision	Recall	F1	Accuracy
Extra Tree	0.019	0.021	0.019	0.004
Random Forest	0.029	0.018	0.021	0.003
Gaussian Naive Bayes	0.014	0.050	0.011	0.024
Multi-layer Perceptron	0.056	0.036	0.040	0.002
SVC (kernel=rbf)	0.048	0.013	0.020	0.001
Logistic Regression	0.077	0.008	0.014	0.001
Stochastic Gradient Descent Classifier	0.045	0.164	0.068	0.061
Perceptron	0.065	0.227	0.095	0.084
Passive Aggressive	0.134	0.242	0.085	0.178
Decision Tree	0.016	0.020	0.017	0.003
K-nearest Neighbors	0.033	0.016	0.020	0.002
Radius Neighbors	0.038	0.014	0.020	0.002
Linear Discriminant Analysis	0.074	0.008	0.014	0.001
Quadratic Discriminant Analysis	0.016	0.090	0.019	0.043
SVC (kernel=sigmoid)	0.018	0.027	0.026	0.009
SVC (kernel=linear)	0.000	0.000	0.000	0.000
Dummy Classifier (strategy=stratified)	0.012	0.013	0.011	0.004
Dummy Classifier (strategy=most frequent)	0.000	0.000	0.000	0.000