Benchmarking Stacking Against Other Heterogeneous Ensembles in Telecom Churn Prediction

Jan Kunnen
research centre for information systems engineering (LIRIS)
KU Leuven
Leuven, Belgium
jan.kunnen@student.kuleuven.be

Maxime Duchateau
research centre for information systems engineering (LIRIS)
KU Leuven
Leuven, Belgium
maxime.duchateau@student.kuleuven.be

Ziboud Van Veldhoven
research centre for information systems engineering (LIRIS)
KU Leuven
Leuven, Belgium
0000-0001-6013-7437

Jan Vanthienen
research centre for information systems engineering (LIRIS)
KU Leuven
Leuven, Belgium
0000-0002-3867-7055

***Abstract*— Customer churn prediction using data mining is an increasingly important concern in the highly saturated telecommunication sector. Albeit its popularity in research, few studies investigated the use of heterogeneous ensembles for this purpose. Therefore, this study evaluated and compared the performance of five grid-searched optimized base classifiers (logistic regression, decision trees, K-nearest neighbors, multilayer perceptron, and support vector machines) and their heterogeneous ensembles (stacking, grading, majority voting, weighted majority voting, and soft voting) on four different telecom datasets. The results indicate that there are significant improvements when using heterogeneous ensembles compared to single classifiers, with stacking being the most performant ensemble. The best meta-classifier for stacking was found to be a multilayer perceptron. Additionally, we identified that using probabilities as an input to the ensemble's meta-classifier, such as in soft voting and stacking variants, can increase their performance.**

Keywords—customer churn prediction, ensembles, data mining, stacking

I. Introduction

For telecommunication businesses, as for most service-providing businesses, increasing revenues and profits often entails growing the customer base. To accomplish this, a company must invest in recruiting new customers and retaining current customers. The difficulty with recruiting new customers in the telecom industry is that the market is saturated, meaning that most people already have a telecom subscription. Drawing in brand new customers usually implies luring them away from their current provider. Studies have shown that attracting new customers can cost up to six times more than retaining existing ones [1]. Customer retention is not only a cost-effective strategy but in addition, can be linked to different loyalty benefits associated with long-term client relationships. For instance, satisfied customers are more likely to engage in word-of-mouth advertising for the company [2]. Therefore, many telecommunication companies have realized that in addition to recruiting new customers, they need to retain their current customers [3]. This is where customer churn prediction (CCP) becomes a valuable tool.

CCP's purpose is to detect customers with a high tendency to leave the company. The customers are divided into two types: (i) customers who churn, *i.e.* stop using the services provided by the company, and (ii) non-churners who continue doing business with the company. To make this distinction, a prediction model is needed that can classify customers as 'churners' based on pattern recognition in the meta-data of the customer's usage and contract data. A clear distinction between these groups allows marketing resources to be more effectively used for retaining doubting customers.

Since the 1990s, ensemble methods have been explored in various fields [4]. However, the use of these models in churn predictions for telecom companies has not yet been thoroughly investigated [5]. Although some ensemble methods have already been tested for this purpose, until recently, a notable absentee was the use of heterogeneous ensembles such as stacking and voting [6], [7]. For this reason, together with the substantial business relevance previously mentioned, we aim to contribute to this field of research by evaluating different heterogeneous ensembles in detail for telecom CCP. This paper mainly focuses on combination methods for single classifiers rather than ensemble compositions. To compare the performance of these ensembles, we introduce the following research questions:

1. *Which meta-algorithm is most suited for combining the single learners in a stacking architecture?*
2. *Do the alternate versions of stacking (probability input, train meta-learning using complementary original attributes) have any merit?*
3. *Do the heterogeneous ensembles have a performance advantage in comparison with the single classifiers?*
4. *Do some heterogeneous ensembles outperform other ensembles?*

In what follows, an overview of the relevant literature will be given, as well as an introduction to many of the models and performance metrics used in this study. The methodology, data, and preprocessing will be explained in the third section followed by the construction of the classifiers. The obtained results, which indicate several benefits of using ensembles over single

978-1-7281-2547-3/20/$31.00 ©2020 European Union

Authorized licensed use limited to: Thompson Rivers University Library. Downloaded on May 26,2024 at 10:08:21 UTC from IEEE Xplore. Restrictions apply.

classifiers, are shown in section four and discussed in section five. The conclusion is given in section six.

II. Background

A. Ensemble Learners

Within telecom CCP, numerous types of single classifiers have been used such as K-nearest-neighbors (KNN), artificial neural networks (ANN) such as multilayer perceptron classifiers (MLPC), support vector machines (SVM), decision trees (DT), and logistic regression (LR) [8]. However, the classification accuracy of single classifiers can be improved by using ensemble methods [9].

In an ensemble, also called a multiple classifier system, multiple base or single learners are trained on the same data after which their predictions are combined. This way, the strengths of the single learners are combined while evening out their weaknesses. Additionally, ensembles tend to be more resilient to noise compared to a single classifier [10]. The goal of ensemble methods is to improve the robustness as well as the performance. For an ensemble to work, there must be diversity in the misclassified instances. There are four ways to accomplish this diversity [11]:

1. *Using different training examples to train individual classifiers (e.g., bagging or boosting).*
2. *Using different training parameters.*
3. *Using different features to train the classifiers (e.g., random subspace method).*
4. *Combining different types of classifiers.*

Ensembles can be divided into heterogeneous and homogeneous ensembles. An ensemble is homogeneous if only one type of single learner is used [12]. This paper focuses on heterogeneous ensembles by combining different single classifiers in the construction of the ensemble (i.e. point 4). In the following pages, a description of frequently used heterogeneous ensembles is given.

B. Heterogeneous Ensembles

The results of the different classifiers used in an ensemble need to be summarized into one final result. There are different methods to do this. With voting, every single classifier chooses a class for each instance independently, then a voting scheme is used to determine the class of each instance [13]. There are three ways in which a class is assigned by the ensemble: (i) unanimous voting (when all classifiers agree); (ii) simple majority (when at least one more than half the number of classifiers agree); or (iii) plurality voting or majority voting (MV) (when the class receives the majority of votes) [11]. The majority voting in a classification task means taking the mode as the prediction of the ensemble. The main drawback of majority voting is the assumption of equally reliable classifiers.

If there is reason to expect that certain classifiers are more qualified than others, weighted majority voting (WV) can improve the performance compared to majority voting. Here, the decisions of more qualified classifiers are assigned higher weights, which are in proportion to the estimated future performance of the classifiers. This estimation can be based on the classifier's performance on a separate validation dataset or the training dataset. The performance depends on the reliability of the estimated accuracies of the classifiers [11].

The voting variants mentioned above are also called hard voting. In the case of soft voting (SV), the classification decision is made based on the average of the churn probabilities predicted by individual classifiers. Therefore, this method only works using classifiers that can calculate the probability of churning [14].

Stacking or stacked generalization is a different technique of combining multiple classifiers. It was introduced by Wolpert in 1992 [15] and has shown to be very performant in the field of classification, especially when the classifiers only tend to be accurate in specific areas of the dataset [16]–[18]. Stacking consists of two phases. First, the base or single learners (level 0) are trained. The output of those models is collected in a new dataset together with the real value that it is supposed to predict. This new dataset is used as input for the stacking model learner (level 1), also known as meta-learner, that combines them and provides the final output. the case of a classification problem, this is also referred to as the meta-classifier [11], [12], [19].

Besides the choices that must be made regarding algorithms and parameters, there is a variety of subtle adaptations that can be made to the ensemble. These options result in a very versatile ensemble but they make analyses and comparisons more difficult since there is much variation in the application of the basic idea [19]. In the following subsections, some examples of different forms of stacking are given.

As mentioned before, an unresolved doubt in stacking is what algorithm is most suited as a meta-classifier. Regarding what classifiers could be used for combining the predictions of the base learner, Wolpert [15] suggested the following: "it is reasonable that relatively global, smooth level-1 generalizers should perform well" (p. 256). Witten et al. [19] also claim that since the base learners already perform some calculus, a simple algorithm such as a linear model or a decision tree with linear models at the leaves should suffice. Some authors, such as Idris and Khan, follow this advice by using a decision tree as a meta-model [20]. However, other authors experimented with alternative meta-classifiers. Examples are the multi-response linear regression [21] or the meta-decision tree where the leaf nodes contained the name of the base classifier that must be used [22]. Furthermore, Torres-Sospedra, Hernández-Espinosa, and Fernández-Redondo proposed to use an ANN to combine the base classifiers [23].

Since most classifiers can also output probabilities for every class, it is possible to set up a stacking architecture where the meta-classifier receives the probabilities as input. The advantage of this method is that the meta-classifier is aware of the confidence of the base classifier, which strengthens inter-level communication [19]. Ting and Witten have shown that using class probabilities can increase performance [21].

Stacking in its original form only passes the predictions of the base classifiers to the meta-learner. However, other research suggests using the base classifiers' outcomes as a complimentary instead of an additional attribute. Chan and Stolfo [24] proposed a class-attribute-combiner, in which the

Authorized licensed use limited to: Thompson Rivers University Library. Downloaded on May 26,2024 at 10:08:21 UTC from IEEE Xplore. Restrictions apply.

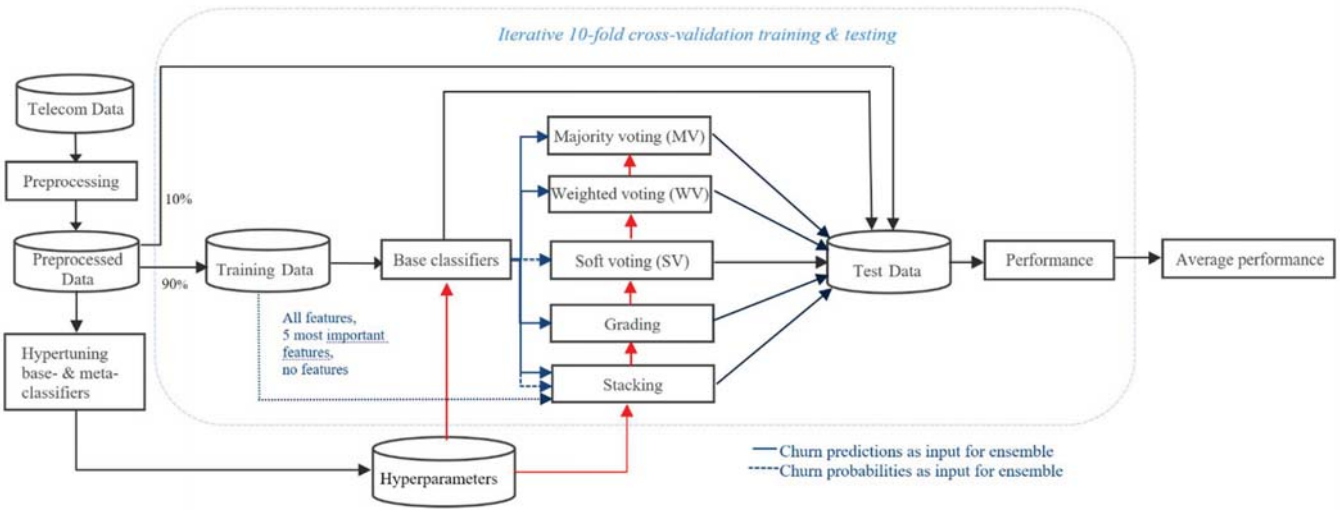

Figure 1: Applied Methodology

original attributes were also fed to the meta-learner. A similar architecture was constructed with Stacked Generalization Plus that outperformed the original stacking in most ensembles [23].

Seewald and Fürnkranz discussed the grading ensemble [25]. Similarly to stacking, this technique uses meta-classifiers to determine the adequateness of the base classifiers. However, in grading each base classifier has its meta-classifier that 'grades' it. The goal of the level-1 learner is to predict the correctness of the base classifier. As input, the original data-attributes and a binary variable indicating whether the base-learner was correct or not are fed into the system. The generated output is the estimated probability that the base classifier is correct. This probability is then used to determine the weight of a base classifier in the weighted voting to reach a final prediction [25]. This modus operandi of selecting the best model by determining for every instance which base learners are most reliable makes grading a hybrid between stacking and purely selecting the best base classifier. It applies multiple nearest neighbors-algorithms that use the original attributes to predict whether a base-learner will be correct [26].

III. Methodology

A. Dataset

For this study, we used four real-world telecommunication datasets. Each dataset includes features on usage, customer behavior, and contract characteristics, such as average call minutes, customer service calls, periodic number of complaints, customer region, and so forth. Since all datasets are labeled, only supervised single classifiers have been used. Involuntary churning data has been disregarded. The main characteristics of the datasets before and after data cleaning are depicted in Table 1.

TABLE 1: DESCRIPTION OF USED DATASETS; CLEANED (ORIGINAL)

Dataset	Cleaned (Original)		
	Instances	*Features*	*Churn rate*
Duke	12,499 (12,499)	9 (11)	39.32% (39.32%)
Chile	7,045 (7,056)	38 (46)	29.07% (29.14%)
UCL	5,000 (5,000)	15 (18)	14.14% (14.14%)
Korea	13,436 (14,490)	40 (20)	23.53% (23.11%)

B. Preprocessing

In the first step, we cleaned the data by manually identifying incorrect or incomplete parts of the data and adjusting or removing these. Duplicated instances and irrelevant features (e.g., client-number and end-date) were removed. Following [27], outliers were handled manually by plotting each attribute. When a cluster of outliers represented more than 5% of the data, no manipulations were conducted. Otherwise, the instances that contained illogical or standalone outliers were removed. Additionally, we removed one of two highly correlated features using a threshold of 90% correlation [28].

Missing values were dealt with based on the missing proportion of instances for each attribute. If less than 5% of values of an attribute were missing, the corresponding instances were removed. If the amount of missing data was between 5% and 30% imputations with the mean-value were used for numerical data. For non-numeric data, the imputation was done with the mode. In the case of over 30% missing values, the attribute was removed from the dataset [27], [29].

Numeric attributes were standardized by applying a min-max scaler where the highest and lowest attribute values were respectively turned into one and zero, and the remaining values were linearly scaled accordingly [30]. This prevents features with large values from dominating other features and can reduce the running time [31]. Finally, all categorical attributes were transformed into dummy variables.

When considering CCP in telecommunication, it appears that the large dimensionality of the datasets makes it difficult for

Authorized licensed use limited to: Thompson Rivers University Library. Downloaded on May 26,2024 at 10:08:21 UTC from IEEE Xplore. Restrictions apply.

both single and ensemble classifiers to achieve the desired performance [32]. This leads to high processing times and low predictive power. Since it was noticed that dummy encoding resulted in around 20 low-frequency categorical features in the Korea dataset, we merged these low-frequency features together with less common regions into one remainder category. This category only represents 3% of the instances.

Customer churn datasets typically have a highly skewed class distribution, which has a negative influence on the performance of CCP models [5], [33]. The datasets used in this paper, i.e. Duke, Chile, UCL, and Korea, have churn rates of respectively 39.32%, 29.07%, 14.14%, and 23.53%. Since a threshold of 12.5% is typically used for class imbalance manipulations, no further corrections were performed.

C. Base Classifiers

The following single or base classifiers are used in this study: LR, KNN, multilayer perceptron classifier (MLPC), support vector machine classifier (SVC), and DT. Before building the classifiers, the optimal hyperparameters were searched. Since an examination into the effects of various kinds of hyperparameter tuning is outside the scope of this paper, an exhaustive grid search has been employed as it is a widely used method for parameter optimization [34]. For evaluating the hyperparameters, the area under the curve (AUC) metric was selected. The detailed description of the parameters can be found through the link in the appendix section VII (cf. A.1, A.2, A.3, and A.4).

D. Ensemble Construction

After creating and training all base classifiers with their optimal hyperparameters for each dataset, they were combined into various heterogeneous ensembles. In majority voting, the final output is the mode of the predictions made by the base classifiers. For weighted majority voting, voting weights were assigned in compliance with the approach used in the homogeneous ensemble Adaboost. Classifier_i was assigned a weight of $\log(1/\beta_i)$, where $\beta_i = \varepsilon_i /(1-\varepsilon_i)$ and ε_i was the weighted training error of classifier_i [11]. We also included soft voting where the prediction is made by rounding the average of the probabilities to churn assigned by the classifiers for each instance. The implementation of grading follows the example of Seewald and Fürnkranz [25]. The meta-learner was a ten nearest neighbor, and the probability produced by the "grader" determined the weight before weighted voting decides the final prediction. For stacking, we constructed and compared 30 variations to make all possible combinations of the options listed below:

1) All single classifiers (LR, KNN, ANN, SVM, and DT) as meta-classifier

2) probabilities vs. binary predictions as inputs for the meta-classifier

3) all, none, and the five most significant original (after preprocessing) attributes used as additional input for the meta-learner

Finally, all constructed ensembles and base classifiers were compared in terms of accuracy, F-measure, AUC, and lift. To add validity to our statements on the performance scores, 10-fold cross-validation, and the average performance was used for more stable results. Furthermore, we tested for significance using a non-parametric McNemar test [6], [35]. An overview of the applied methodology is shown in Fig. 1.

IV. Results

A. Combining Base Learners in a Stacking Architecture

To answer our first research question regarding the most suited meta-learner for the stacking architecture, we compared the performance of all stacking variants (cf. A.5). We found that the performance of the stacking models is quite similar regardless of the meta-classifier used. However, MLPC as meta-classifier appears to slightly outperform the other classifiers when considering all datasets and performance metrics. MLPC performs best in terms of AUC and F1 and is a close second to KNN in terms of accuracy. Looking at each dataset separately, we can see that MLPC is the best performing meta-classifier for all datasets except for the UCL, where the KNN does seem to outperform the others slightly. However, the use of KNN in the stacking architecture tends to lead to heavy overfitting, bringing forth high accuracies, AUC, and F1 scores for the training sets of all datasets except for Korea.

B. Alternate Versions of Stacking (Probability Input, Train Meta-learning Using Complementary Original Attributes)

Next, the relative performances of the alternate versions of stacking are evaluated, providing an answer to our second research question (cf. A.5 and the significance tests in A.9). The overall results clearly show that the use of probabilities in the stacking architecture instead of predicted binary outcomes improves the performance. Ceteris paribus using probabilities instead of predictions results in an average improvement in terms of accuracy, AUC, F1, and lift of respectively 0.8%, 4.2%, 3.2%, and 6.5% (cf. A.7). The same trend is visible in probability-stacking ensembles where they make up the top five in three datasets, mostly significantly outperforming the best prediction-stacking ensemble. This indicates that, as expected, there is merit in using the probability-approach. Furthermore, stacking using only predictions results has the worst performance in terms of all performance metrics with only some exceptions and never seems performant in comparison to the other variants (cf. A.7). The best performing prediction-only stacking ensembles still significantly underperform compared to all probability varieties and all other stacking ensembles. However, when predictions instead of probabilities are used as input for the SVC meta-classifier, the performance can increase, with the most extreme shift in the Korea dataset where the F1 performance increases by 79.1%.

Next, we investigated whether providing a set of original features as additional inputs to the meta-classifier is advantageous (cf. A.8). The performance gains here are less noticeable. There does not seem to be a clear benefit except for the AUC and lift scores when predictions are used: there is an increase in performance of 5.3% AUC and a 4.8% lift when using all attributes. AUC and lift rose 1.8% and 1.2% respectively when only including the five most important ones. Therefore, it is not surprising that the McNemar tests also do not show significant improvements (cf. A.9).

Authorized licensed use limited to: Thompson Rivers University Library. Downloaded on May 26,2024 at 10:08:21 UTC from IEEE Xplore. Restrictions apply.

TABLE II: PERFORMANCE OF BASE CLASSIFIERS AND HETEROGENEOUS ENSEMBLES

CHILE	LR	DT	KNN	MLPC	SVC	MV	WV	SV	Grading	Stacking*
Accuracy	0.822	0.861	0.853	0.900	0.866	0.880	0.880	0.890	0.880	**0.905**
AUC	0.878	0.924	0.904	0.964	0.931	0.912	0.915	0.958	0.915	**0.967**
F1	0.633	0.742	0.696	0.821	0.741	0.762	0.762	0.788	0.764	**0.836**
Lift	2.715	2.763	2.997	2.942	2.909	**3.096**	**3.096**	3.095	3.090	2.889

DUKE	LR	DT	KNN	MLPC	SVC	MV	WV	SV	Grading	Stacking*
Accuracy	0.952	0.955	0.950	0.954	0.957	0.957	0.957	0.959	0.958	**0.967**
AUC	0.974	0.981	0.984	0.974	0.972	0.965	0.965	0.982	0.965	**0.987**
F1	0.936	0.941	0.932	0.939	0.942	0.943	0.943	0.945	0.944	**0.957**
Lift	2.482	2.490	2.534	2.505	2.520	2.524	2.535	**2.536**	2.524	2.510

KOREA	LR	DT	KNN	MLPC	SVC	MV	WV	SV	Grading	Stacking*
Accuracy	0.764	0.766	0.767	0.773	0.773	0.776	0.776	0.776	**0.780**	0.773
AUC	0.622	0.667	0.646	0.673	0.655	0.593	0.593	**0.694**	0.596	0.691
F1	0.005	0.232	0.149	0.265	0.162	0.139	0.139	0.127	0.180	**0.333**
Lift	1.096	2.156	2.268	2.379	2.598	**3.132**	**3.132**	3.315	3.073	2.279

UCL	LR	DT	KNN	MLPC	SVC	MV	WV	SV	Grading	Stacking*
Accuracy	0.865	0.935	0.870	0.876	0.924	0.893	0.893	0.913	0.905	**0.953**
AUC	0.823	0.894	0.887	0.858	0.902	0.881	0.884	**0.913**	0.886	0.905
F1	0.266	0.752	0.170	0.413	0.677	0.434	0.434	0.581	0.525	**0.820**
Lift	4.074	5.798	6.293	4.430	5.992	5.981	5.981	**6.430**	6.273	6.307

*Stacking variant using MLPC as meta-classifier and probabilities in combination with the five most important features in terms of recursive feature elimination using 10-fold cross-validation as input.
LR=logistic regression, DT=Decision Tree, KNN = K-Nearest-Neighbors, MLPC= multilayer perceptron, SVC = support vector machine classifier, MV = Majority Voting, WV = Weighted Voting, SV = Soft Voting

Although all stacking set-ups using probabilities as inputs perform well and similarly, we found the best overall performing model using MLPC as meta-classifier and probabilities in combination with the five most important features (according to rfecv) as input (cf. A.5). Although this does not always result in the highest attainable performance score, it performs consistently: it raises the average accuracy, AUC, F1, and lift by 0.1%, 6.7%, 7%, and 4.9% respectively relative to stacking with predictions as the only input. Therefore, we will use this type to answer subsequent research questions where it will represent the stacking ensembles when comparing them to single learners and other heterogeneous ensembles.

C. Heterogeneous Ensembles vs. Base Classifiers

To answer the last two research questions, we refer to Table 2 where we summarize the performance of all single classifiers and their ensembles tested in this paper. The bold font indicates the best performance scores. It is apparent from this table that heterogeneous ensembles have a performance benefit compared to base classifiers, giving an answer to research question three. When we make the comparison between the most performant stacking ensemble (MLPC as meta-classifier and probabilities in combination with the five most important features) and the best performing base classifier for that specific dataset and performance metric, the average performance improvement in terms of accuracy, AUC, F1 and lift are respectively 1.1%, 1.2%, 9.5%, and 4.6%. These results gain validity due to the 10-fold cross-validation (cf. A.9). It is apparent from Table 2 that in every dataset at least one ensemble is substantially more performant than the best base classifier. Apart from Korea, where grading excels, the five most performant algorithms according to the McNemar test are all stacking ensembles.

Although there is a clear distinction, we note that some of the base classifiers (e.g., MLPC) are also very performant while being computationally less intensive. The performance advantage mainly exists for the more complex heterogeneous ensembles (stacking and grading). However, soft voting also

Authorized licensed use limited to: Thompson Rivers University Library. Downloaded on May 26,2024 at 10:08:21 UTC from IEEE Xplore. Restrictions apply.

performs well, obtaining the highest AUC scores twice. The lift metric seems to lean more towards the voting ensembles.

D. Performance of Heterogeneous Ensembles

For the final research question on the comparison of all heterogeneous ensembles, the cross-validated results show that there are considerable differences in performance among the different heterogeneous ensembles. Table 2 indicates the dominance of the MLPC stacking variant over the other heterogeneous ensembles. The grading ensemble, which is one of the more complex heterogeneous ensembles, tends to underperform compared to the other ensembles except for the high accuracy in the Korea dataset, which has both the largest number of instances and features.

When focusing solely on the voting variants, two important remarks can be made. First, majority voting and weighted majority voting return the same 10-fold cross-validated performance in terms of accuracy and F1 and similar AUC scores. Secondly, when comparing majority voting and weighted majority voting to the other heterogeneous models, it is apparent that both show inferior performance on all datasets and all performance measures except lift. The soft voting classifier outperforms the other voting variants in all datasets except for a slight underperformance in the Korea dataset. These findings are also significant (cf. A.9). The average performance improvement of soft voting compared to the classical majority voting is 0.9% accuracy, 6.8% AUC, 7.2% F1, and 3.4% lift. Compared to the other heterogeneous ensembles, soft voting performs relatively well, especially in terms of AUC where it is the best performing classifier in two datasets.

Regarding the choice between voting and stacking our research indicates that stacking is more performant. In three out of four datasets all but a few stacking-ensembles are significantly more accurate than voting set-ups (cf. A.9).

V. Discussion

This study aimed to determine the potential of different heterogeneous ensembles compared to single classifiers in the context of CCP. It included majority voting, weighted voting, soft voting, grading, and numerous adaptations of stacking. For building and contrasting these ensembles, five base classifiers were used: LR, DT, KNN, MLPC, and SVC. The performance was compared using accuracy, AUC, F1-score, and lift.

Our findings indicate that there is some merit in using more complex heterogeneous ensembles over single classifiers. This can be a simple combination rule, such as soft voting or a more complex stacking set-up. This is in line with the conclusions in [5] on stacking outperforming base learners. Another important finding is the positive impact of using probabilities instead of predictions for voting and stacking. From a theoretical perspective, a possible explanation for this is that using probabilities strengthens inter-level communication. Nevertheless, the use of predictions instead of probabilities is still common for voting and stacking in the literature on churn predictions in the telecommunication sector [5], [20]. Since soft voting's performance overshadows classical voting and does not require additional computing power or technical skills, it does provide an interesting alternative compared to the more commonly used base classifiers [14]. Further research should be undertaken to investigate this. The results suggest that stacking has the potential to outperform voting. These findings contradict [20] while confirming Džeroski and Ženko's findings [36]. However, our results dispute their statement saying merely selecting the best performing base classifier outperforms stacking.

Regarding the selection of the meta-classifier, we advise using the MLPC classifier due to its strong and consistent performance. Concerning the benefits of providing a set of original attributes as an additional input to the meta-classifier, our conclusion is less outspoken. Although there does seem to be a small increase in performance, this technique increases the dimensionality and therefore the computing-requirements. This makes it harder to justify its adaption. These findings are in line with the findings in [36], stating that an extended set of meta-level attributes does not imply clearly better performance. They suggest that the advantage of providing additional information is countered by the adverse effect of an increased dimensionality of the meta-data.

What also stands out is that weighted voting and majority voting seldom disagree on a prediction. This is not unexpected since the variance in the weights used was small, following [11]. Alternative methods of weight determination may be more performant. Another important remark is that the divergence between lift and F1 in Table 2 is due to the nature of these metrics. The low recall-rate in the Korea-dataset lowers the F1-score drastically while the lift-measure does not consider the recall. Therefore it is logical that these measures disagree over the performance.

The limitations of this study include the relatively small dataset samples. It is possible that these findings cannot be generalized for more complex datasets, which can be a future research avenue. Although we tuned our hyperparameters using a thorough grid search (cf. A.1 to A.4), slight differences can be obtained by using different hyperparameters. Another constraint is that this work only used a small sample of all classifiers and performance metrics available. Notable absentees are the more complex homogeneous ensembles. A comparison between these and the used heterogeneous ensembles could be a fruitful area for further research. Lastly, it is unclear whether the large dimensionality plays a role in grading's excellent performance. Further research would be required to investigate this relationship.

VI. Conclusion

This study set out to explore the usage of heterogeneous ensembles in CCP. The results show the relevance of heterogeneous ensembles, returning superior performance in comparison to single classifiers. This is true for simple voting models as well as the more complex stacking ensembles that obtain the highest performance across all tested classifiers. Another finding is the significant impact of using probabilities as input to heterogeneous ensembles instead of the commonly used predictions. Our grading model also showed great promise when applied to a highly dimensional dataset. Further research should be undertaken to explore the benefits of ensembles for commercial use.

Authorized licensed use limited to: Thompson Rivers University Library. Downloaded on May 26,2024 at 10:08:21 UTC from IEEE Xplore. Restrictions apply.

VII. Appendix

Due to page limitations, the detailed description of the classifiers, parameters, and results of this paper can be accessed through the following link: https://feb.kuleuven.be/ziboud.vanveldhoven/public/AUSDM2020/appendix. The grid search results are summarized in A.1 to A.4. The performance of the stacking variants are shown in A.5, the weight determination for the weighted majority voting in A.6, the improvement of using probabilities in the stacking variants in A.7, the average improvement of using extra attributes for meta-classifiers in A.8, and the results of the McNemar's test in A.9.

References

[1] C. B. Bhattacharya, "When customers are members: Customer retention in paid membership contexts," *J. Acad. Mark. Sci.*, vol. 26, no. 1, pp. 31–44, 1998.

[2] J. Ganesh, M. J. Arnold, and K. E. Reynolds, "Understanding the customer base of service providers: an examination of the differences between switchers and stayers," *J. Mark.*, vol. 64, no. 3, pp. 65–87, 2000.

[3] A. D. Athanassopoulos, "Customer satisfaction cues to support market segmentation and explain switching behavior," *J. Bus. Res.*, vol. 47, no. 3, pp. 191–207, 2000.

[4] Z.-H. Zhou, *Ensemble methods: foundations and algorithms*. CRC press, 2012.

[5] H. Abbasimehr, M. Setak, and M. J. Tarokh, "A comparative assessment of the performance of ensemble learning in customer churn prediction.," *Int. Arab J. Inf. Technol.*, vol. 11, no. 6, pp. 599–606, 2014.

[6] A. Idris, A. Iftikhar, and Z. ur Rehman, "Intelligent churn prediction for telecom using GP-AdaBoost learning and PSO undersampling," *Cluster Comput.*, vol. 22, no. 3, pp. 7241–7255, 2019.

[7] B. Zhu, B. Baesens, A. Backiel, and S. K. L. M. Vanden Broucke, "Benchmarking sampling techniques for imbalance learning in churn prediction," *J. Oper. Res. Soc.*, vol. 69, no. 1, pp. 49–65, 2018.

[8] T. Vafeiadis, K. I. Diamantaras, G. Sarigiannidis, and K. C. Chatzisavvas, "A comparison of machine learning techniques for customer churn prediction," *Simul. Model. Pract. Theory*, vol. 55, pp. 1–9, 2015.

[9] M. K. Awang, M. Makhtar, M. N. Abd Rahman, and M. M. Deris, "A new customer churn prediction approach based on soft set ensemble pruning," in *International Conference on Soft Computing and Data Mining*, 2016, pp. 427–436.

[10] I. Syarif, E. Zaluska, A. Prugel-Bennett, and G. Wills, "Application of bagging, boosting and stacking to intrusion detection," in *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, 2012, pp. 593–602.

[11] R. Polikar, "Ensemble based systems in decision making," *IEEE Circuits Syst. Mag.*, vol. 6, no. 3, pp. 21–45, 2006.

[12] M. P. Sesmero, A. I. Ledezma, and A. Sanchis, "Generating ensembles of heterogeneous classifiers using stacked generalization," *Wiley Interdiscip. Rev. data Min. Knowl. Discov.*, vol. 5, no. 1, pp. 21–34, 2015.

[13] P.-N. Tan, M. Steinbach, and V. Kumar, *Introduction to data mining*. Pearson Education India, 2016.

[14] X. Wang, K. Nguyen, and B. P. Nguyen, "Churn Prediction using Ensemble Learning," in *Proceedings of the 4th International Conference on Machine Learning and Soft Computing*, 2020, pp. 56–60.

[15] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[16] K. Kim and E. B. Bartlett, "Error estimation by series association for neural network systems," *Neural Comput.*, vol. 7, no. 4, pp. 799–808, 1995.

[17] M. LeBlanc and R. Tibshirani, "Combining estimates in regression and classification," *J. Am. Stat. Assoc.*, vol. 91, no. 436, pp. 1641–1650, 1996.

[18] L. Todorovski, H. Blockeel, and S. Dzeroski, "Ranking with predictive clustering trees," in *European Conference on Machine Learning*, 2002, pp. 444–455.

[19] I. H. Witten and E. Frank, "Data mining: practical machine learning tools and techniques with Java implementations," *Acm Sigmod Rec.*, vol. 31, no. 1, pp. 76–77, 2002.

[20] A. Idris and A. Khan, "Churn prediction system for telecom using filter–wrapper and ensemble classification," *Comput. J.*, vol. 60, no. 3, pp. 410–430, 2017.

[21] K. M. Ting and I. H. Witten, "Issues in stacked generalization," *J. Artif. Intell. Res.*, vol. 10, pp. 271–289, 1999.

[22] L. Todorovski and S. Džeroski, "Combining multiple models with meta decision trees," in *European conference on principles of data mining and knowledge discovery*, 2000, pp. 54–64.

[23] J. Torres-Sospedra, C. Hernández-Espinosa, and M. Fernández-Redondo, "Combining MF networks: A comparison among statistical methods and stacked generalization," in *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, 2006, pp. 210–220.

[24] P. K. Chan and S. J. Stolfo, "A comparative evaluation of voting and meta-learning on partitioned data," in *Machine Learning Proceedings 1995*, Elsevier, 1995, pp. 90–98.

[25] A. K. Seewald and J. Fürnkranz, "An evaluation of grading classifiers," in *International symposium on intelligent data analysis*, 2001, pp. 115–124.

[26] J. Burez and D. Van den Poel, "Handling class imbalance in customer churn prediction," *Expert Syst. Appl.*, vol. 36, no. 3, pp. 4626–4636, 2009.

[27] W. Verbeke, K. Dejaeger, D. Martens, J. Hur, and B. Baesens, "New insights into churn prediction in the telecommunication sector: A profit driven data mining approach," *Eur. J. Oper. Res.*, vol. 218, no. 1, pp. 211–229, 2012.

[28] E. Stripling, S. vanden Broucke, K. Antonio, B. Baesens, and M. Snoeck, "Profit maximizing logistic regression modeling for customer churn prediction," in *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, 2015, pp. 1–10.

[29] J. Basiri, F. Taghiyareh, and B. Moshiri, "A hybrid approach to predict churn," in *2010 IEEE Asia-Pacific Services Computing Conference*, 2010, pp. 485–491.

[30] T. Rashid, "Classification of Churn and non-churn customers for telecommunication companies," *Int. J. Biometrics Bioinforma.*, vol. 3, no. 5, pp. 82–89, 2010.

[31] P. Baumann, D. S. Hochbaum, and Y. T. Yang, "A comparative study of the leading machine learning techniques and two new optimization algorithms," *Eur. J. Oper. Res.*, vol. 272, no. 3, pp. 1041–1057, 2019.

[32] A. Idris and A. Khan, "Customer churn prediction for telecommunication: Employing various various features selection techniques and tree based ensemble classifiers," in *2012 15th International Multitopic Conference (INMIC)*, 2012, pp. 23–27.

[33] Y. Wang and J. Xiao, "Transfer ensemble model for customer churn prediction with imbalanced class distribution," in *2011 International Conference of Information Technology, Computer Engineering and Management Sciences*, 2011, vol. 3, pp. 177–181.

[34] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *J. Mach. Learn. Res.*, vol. 13, no. 1, pp. 281–305, 2012.

[35] A. Amin *et al.*, "Comparing oversampling techniques to handle the class imbalance problem: A customer churn prediction case study," *IEEE Access*, vol. 4, pp. 7940–7957, 2016.

[36] S. Džeroski and B. Ženko, "Is combining classifiers with stacking better than selecting the best one?," *Mach. Learn.*, vol. 54, no. 3, pp. 255–273, 2004.

Authorized licensed use limited to: Thompson Rivers University Library. Downloaded on May 26,2024 at 10:08:21 UTC from IEEE Xplore. Restrictions apply.