Machine Learning

CSI 5155

Semi-Supervised Learning, Label Scarcity, and Class Imbalance

Course Project

Authors
Alexandra Sklokin (300010511)

Professor
Dr. Herna Viktor

Contents

1 Introduction

The purpose of this project is to learn about different semi-supervised classification methods, as well as their interplay with skewed data. Then I will try undersampling and oversampling to determine the effect on semi-supervised learning. I will use six levels of unlabelled data 0%, 10%, 20%, 50%, 90%, and 95%. Thus, I will be constructing 162 semi-supervised models (three semi-supervised classification methods, three datasets, three resampling methods and six levels of label scarcity).

The following report documents my methodology, results, and leasons learned for this project.

2 Methodology

For this project I used *jupyter notebooks*, Excel spreadsheets, and the *sklearn Python* library. Please find all of the code for this project in my git repository[1].

3 Datasets

Our analysis will be done on three datasets, *Online_Shopping_Intention*, *Marketing_Campaign*, and *Heart* (*Table 1*). Original data can be found in the *data/original* folder in my git repository.

Dataset	Features	Instances	target=0	target=1
Online_Shopping_Intention	17	12330	10422 (84.5%)	1908
Marketing_Campaign	25	2240	1158 (51.7%)	1082
Heart	13	303	165 (54.5%)	138

Table 1: Size and Distribution of Datasets

4 Preprocessing

Preprocessing of data is one of the most important Machine Learning steps. In fact, taking care of data makes up about 70% of the effort. This was certainly true for the project.

4.1 Cleaning the Data

First, cleaning the data. For the *Online_Shopping_Intention* and *Heart* datasets, I did not need to do any cleaning. For *Marketing_Campaign*, I (1) filled all empty

[1] https://github.com/alexandrasklokin/CSI5155/tree/main/Project

Income values with the column average, (2) remapped *Marital Status* to account for absurd values (ex. *Yolo* to *Unknown*), (3) transformed *DT_Customer* to number of days since the first day and (4) removed *Z_CostContact*, *Z_Revenue* and *ID* features since the values are either constant or always unique. To make sure there is no inherent ordering in the data, I shuffled and re-indexed all of the datasets.

4.2 Feature Transformation

For categorical features (features with discrete or non-numerical values), I used Ordinal Encoding, where non-numerical values in each categorical variable are transformed to integers (ex. *month* 'Feb' to 2).

For the target features, I encoded *True* entries to 1, and *False* to 0, so that I could use tools which required numerical based (not string) values.

For numerical features (features with continuous values), I standardized the entries. The entries are scaled so that for each feature the mean of all entries is equal to 0.0, and the standard deviation is 1.0.

4.3 Resampling

Two methods of resampling were performed for this project. Oversampling is when the minority class is resampled to match the number of occurrences of the majority class. Undersampling is when we sample from the majority class, and only take as many entries as in the minority class.

Resampling should be done only in the training set, as in practice, unseen data might follow the observed unbalanced distribution. The testing data should be representative of the true population, thus it is left out during resampling.

The training data, following the cleaning, processing and resampling steps above, can be found in the *data/processed* folder in my git repository. The test data (which was not resampled) is found in *data/test*.

4.4 Unlabelling

For each of the 3 datasets, and 3 resampling methods (no-resampling, undersampling, and oversampling), I produced 6 datasets with varying levels of unlabelled data. The levels were 0%, 10%, 20%, 50%, 90% and 95%. Unlabelling was only done of the training set, since I would like the test set to be labelling for evaluation of model predictions.

The training data, following unlabelling, can be found in the *data/train* folder in my git repository.

5 The Models

5.1 Self-Training

The Self-Training algorithm relies on iteratively training a supervised classifier to pseudo-label unlabelled entries, and add them to the labelled ones. It is part of the Wrapper-based class of semi-supervised methods. This algorithm is implemented in the *sklearn.semi-supervised* library.

5.1.1 Supervised Model Selection

First, I determined which supervised classifier performs best for each dataset and resampling combination (*Table 2*). The classifier is a parameter of the Self-Training algorithm. I compared the F1-Scores for the following models: K-Nearest Neighbours, Support Vector Machine, Decision Tree, Random Forest, Multi Layer Perceptron, and Gradient Boosting Ensemble.

Dataset	No Resampling	Undersampling	Oversampling
Online_Shopping_Intention	GBE	RF	RF
Marketing_Campaign	GBE	RF	RF
Heart	RF	DT	DT

Table 2: Models with Highest F1-Score for each Dataset and Resampling Combination

For each of the above nine supervised models, I performed hyperparameter tuning using *sklearn*'s *GridSearchCV*. The following hyperparameters were obtained:

- **Online_Shopping_Intention**
  GBE: *'learning_rate': 1, 'max_depth': 1, 'n_estimators': 5.* **RF:** *'max_depth': 5, 'max_features': 3, 'min_samples_leaf': 3, 'min_samples_split': 8, 'n_estimators': 100.*

- **Marketing_Campaign**
  GBE: *'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 250.* **RF:** *'max_depth': 15, 'max_features': 5, 'min_samples_leaf': 3, 'min_samples_split': 8, 'n_estimators': 200.*

- **Heart**
  RF: *'max_depth': 5, 'max_features': 2, 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 200.* **DT:** *'criterion': 'entropy', 'max_depth': 2, 'min_samples_leaf': 1, 'min_samples_split': 2.*

5.2 SemiBoost

The SemiBoost algorithm is an example of semi-supervised ensembles, where the final prediction is a linear combination of the base learners of the ensemble. At each iteration, unlabelled instances as assigned a pseudo-label. A subset of the pseudo-labelled instances are selected based on our confidence in their accuracy, and added to the labelled instances, and used to train the next base learner. Each model is given a specific weight, to be used in the final prediction calculation.

5.2.1 Implementation

For the project, I adapted GitHub user *papabloblo*'s implementation of SemiBoost [2]. This user's original code had many bugs when used with my data and data types. I edited the *fit*, *predict*, and *predict_proba* functions. Since there was no *score* function, I was unable to do hyperparameter tuning.

5.3 Label Propagation

The Label Propagation algorithm is an example of transductive methods, utilizing graphs. Each instance is a node, and a graph is constructed using edges between these nodes. At each iteration a soft label assignment is propagated to estimate the labels at each unlabelled instance neighbouring a labelled instance, based on edge weights. Predictions are calculated as the weighted sum of the labels of an instances neighbours. This algorithm is implemented in the *sklearn.semi-supervised* library.

6 Training

6.1 Hyperparameter Tuning

I performed hyperparameter tuning of the Self-Training, and Label Propagation algorithms using *GridSearchCV*, for each dataset. The following hyperparameters were obtained:

- **Online_Shopping_Intention**
  Self Training (GBE): *'criterion': 'threshold', 'k_best': 10, 'max_iter': 200, 'threshold': 0.5.* **Self Training (RF):** *'criterion': 'threshold', 'k_best': 100, 'max_iter': 200, 'threshold': 0.9.* **Label Propagation:** *'gamma': 20, 'kernel': 'knn', 'max_iter': 10, 'n_neighbors': 5, 'tol': 0.0001.*

- **Marketing_Campaign**
  Self Training (GBE): *'criterion': 'threshold', 'k_best': 10, 'max_iter':*

None, 'threshold': 0.8. **Self Training (RF):** *'criterion': 'threshold', 'k_best': 200, 'max_iter': 10, 'threshold': 0.8.* **Label Propagation:** *'gamma': 20, 'kernel': 'knn', 'max_iter': 10, 'n_neighbors': 5, 'tol': 0.0001.*

- **Heart**
  Self Training (RF): *'criterion': 'threshold', 'k_best': 10, 'max_iter': 10, 'threshold': 0.8.* **Self Training (DT):** *'criterion': 'k_best', 'k_best': 10, 'max_iter': 100, 'threshold': 0.9.* **Label Propagation:** *'gamma': 40, 'kernel': 'rbf', 'max_iter': 10, 'n_neighbors': 3, 'tol': 0.0001.*

Please note that during training of the Label Propagation algorithm, some parameters needed to be switched so that the *model.predict_prob()* function did not return *nan*. The algorithm attempts to create a graph, but with the above hyperparameters, it is not able to construct a fully connected graph.

6.2 Prediction

Lastly, I train my three semi-supervised classifiers against 54 different datasets (3 datasets x 3 resampling methods x 6 levels of labelling). Target predictions on the test set can be found in the *pred/selftraining*, *pred/semiboost*, and *pred/labelpropagation* folders in my git repository.

6.3 Evaluation

After training all 162 models I evaluate them by recording F1-scores, accuracies, execution times, AUCs, and ROC curves. I will primarily use F1-score and AUC to evaluate the performance of my models.

Using Friedman tests and the accuracies, I can determine if the three semi-supervised methods are significantly different, for different resampling methods and unlabelling levels. Then using the Nemenyi Critical Difference diagrams, I will be able to select a best model. I used GitHub user *hfawaz*'s implementation of Nemenyi CD Diagrams [1]. In this implementation, thick lines indicate lack of significant difference.

7 Experimental Results

7.1 ROC Curves

Refer to *Appendix A, B, and C* for ROC curves of each classifier. Bigger format can be found in the *roc* folder of my git repository.

7.2 F1-Scores, Accuracies, and Execution Times

7.2.1 Self-Training

Resampling	Unlabelled %	F1-Score	Accuracy	AUC	Time (sec)
Online_Shopping_Intention					
None	0	88.38	88.24	89.54	0.08
	10	77.44	84.51	35.58	0.1
	20	77.45	84.54	42.42	0.1
	50	77.43	84.48	61.21	0.1
	90	78.21	84.16	50.97	0.1
	95	77.46	84.05	49.3	0.08
Under	0	87.74	86.64	92.5	0.67
	10	40.94	35.88	50.43	0.75
	20	22.95	24.38	50.33	0.67
	50	12.91	19.57	60.75	0.66
	90	26.6	26.76	56.06	0.52
	95	41.75	36.74	50.2	0.47
Over	0	87.31	86.16	91.82	1.98
	10	16.35	20.28	42.52	2.77
	20	5.8	16.17	53.84	1.73
	50	4.9	15.79	56.2	1.29
	90	4.19	15.46	52.11	0.7
	95	14.43	17.98	30.58	0.64
Marketing_Campaign					
None	0	91.37	91.37	97.33	1.57
	10	49.75	50.0	50.36	5.26
	20	46.74	46.73	47.0	7.93
	50	51.63	51.64	53.81	25.09
	90	42.2	43.75	43.04	29.14
	95	48.94	49.26	46.87	15.87
Under	0	94.5	94.49	98.79	0.84
	10	47.39	47.47	49.21	2.71
	20	51.88	52.08	53.42	1.52
	50	41.94	43.01	44.12	2.41
	90	40.17	49.55	56.99	5.67
	95	31.53	48.36	56.24	5.55
Over	0	88.84	88.84	95.53	0.77
	10	55.06	55.21	56.87	3.04
	20	53.0	53.57	55.65	2.42
	50	44.35	46.28	48.16	5.48
	90	31.53	48.36	33.89	6.36
	95	37.48	50.0	67.37	5.23
Heart					
None	0	63.95	65.93	70.1	1.3
	10	30.84	30.77	24.2	0.71
	20	80.22	80.22	83.27	0.66
	50	62.78	67.03	78.58	0.79
	90	38.46	53.85	54.73	0.65
	95	38.97	54.94	80.58	0.63
Under	0	31.52	45.06	49.46	0.03
	10	40.4	40.66	44.12	0.02
	20	46.3	48.35	59.85	0.03
	50	38.46	53.85	52.42	0.06
	90	60.76	61.54	60.17	0.06
	95	57.36	60.44	57.85	0.06
Over	0	30.61	40.66	46.63	0.02
	10	53.95	53.85	54.24	0.02
	20	45.31	57.14	45.63	0.03
	50	49.02	49.45	43.32	0.04
	90	60.54	60.44	61.07	0.04
	95	51.52	51.65	50.95	0.06

Table 3: Results for Self-Training Classifier.

7.2.2 SemiBoost

Resampling	Unlabelled %	F1-Score	Accuracy	AUC	Time (sec)
Online_Shopping_Intention					
None	0	89.32	89.86	92.68	33.62
	10	0.2	0.11	49.66	214.52
	20	0.8	0.57	50.0	224.58
	50	4.18	13.06	51.64	278.04
	90	4.15	15.46	46.76	212.44
	95	4.14	15.46	46.74	208.52
Under	0	89.81	88.78	97.22	1.75
	10	3.45	9.38	52.22	20.19
	20	3.49	10.73	56.4	22.27
	50	3.72	13.25	65.76	24.06
	90	4.06	15.06	57.83	22.99
	95	4.09	15.22	53.17	24.75
Over	0	89.76	89.78	92.88	111.85
	10	4.14	15.46	52.61	776.52
	20	4.14	15.46	49.95	635.3
	50	4.14	15.46	47.52	965.95
	90	4.14	15.46	47.27	963.41
	95	4.14	15.46	41.17	924.91
Marketing_Campaign					
None	0	89.59	89.58	96.05	0.87
	10	25.98	29.61	49.16	10.25
	20	30.14	40.48	52.39	10.5
	50	31.48	47.47	55.32	12.38
	90	31.3	47.77	46.9	13.36
	95	31.46	48.21	47.55	20.34
Under	0	96.88	96.88	99.3	0.88
	10	25.41	28.57	49.18	16.44
	20	27.18	33.04	51.95	25.83
	50	29.85	40.77	52.93	22.8
	90	31.25	47.02	57.07	21.69
	95	31.39	47.77	56.65	19.38
Over	0	89.13	89.14	95.56	1.91
	10	30.11	37.2	56.56	14.64
	20	31.23	44.49	50.78	12.27
	50	31.85	48.06	49.34	12.08
	90	31.63	48.36	44.47	14.42
	95	31.66	48.36	54.93	12.85
Heart					
None	0	54.03	54.94	61.39	0.39
	10	24.18	24.18	30.95	2.6
	20	39.52	45.06	70.58	2.89
	50	38.22	52.75	65.95	4.17
	90	38.97	54.94	45.15	3.24
	95	38.97	54.94	70.73	2.9
Under	0	40.59	40.66	38.02	0.24
	10	30.4	28.57	56.24	2.61
	20	31.25	34.07	43.78	2.74
	50	37.5	47.25	35.29	2.8
	90	37.95	52.75	48.22	2.62
	95	38.97	54.94	49.05	2.4
Over	0	56.04	56.04	54.15	0.22
	10	39.05	47.25	60.34	1.81
	20	39.56	49.45	63.95	2.13
	50	36.9	49.45	34.24	2.4
	90	38.97	54.94	56.07	2.5
	95	38.97	54.94	62.68	2.67

Table 4: Results for SemiBoost Classifier.

7.2.3 Label Propagation

Resampling	Unlabelled %	F1-Score	Accuracy	AUC	Time (sec)
Online_Shopping_Intention					
None	0	77.45	84.54	75.45	9.96
	10	77.45	84.54	39.08	10.82
	20	77.45	84.54	51.56	10.36
	50	77.45	84.54	54.83	10.49
	90	77.45	84.54	39.99	14.46
	95	77.45	84.54	36.76	20.78
Under	0	65.78	60.29	73.97	3.68
	10	4.14	15.46	42.73	3.72
	20	4.14	15.46	57.29	3.75
	50	4.14	15.46	62.69	3.97
	90	4.14	15.46	48.02	4.24
	95	4.14	15.46	57.02	6.32
Over	0	77.45	72.97	79.42	16.12
	10	77.45	16.27	41.39	19.06
	20	77.45	15.44	53.22	16.65
	50	77.45	15.46	49.82	17.74
	90	77.45	15.46	60.52	27.49
	95	77.45	15.46	47.76	36.72
Marketing_Campaign					
None	0	74.54	74.55	83.98	0.63
	10	44.59	48.96	47.14	0.49
	20	40.01	42.11	35.7	0.53
	50	56.69	59.52	68.67	0.51
	90	31.53	48.36	31.21	0.72
	95	36.54	41.22	41.43	0.89
Under	0	74.94	75.15	84.29	0.41
	10	50.9	51.04	49.75	0.41
	20	49.57	53.72	62.82	0.43
	50	31.46	48.21	46.0	0.45
	90	31.53	48.36	47.81	0.6
	95	31.53	48.36	54.94	0.65
Over	0	74.54	75.0	84.2	0.5
	10	44.59	43.6	44.96	0.53
	20	40.01	50.89	52.29	0.49
	50	56.69	49.7	45.38	0.45
	90	31.53	48.36	37.35	0.6
	95	36.54	48.36	62.22	0.83
Heart					
None	0	60.51	60.44	61.17	0.04
	10	40.8	40.66	41.61	0.03
	20	58.13	58.24	60.39	0.03
	50	57.85	59.34	60.54	0.03
	90	57.78	58.24	52.61	0.04
	95	64.44	64.83	71.39	0.04
Under	0	44.05	43.96	42.98	0.03
	10	52.75	52.75	53.8	0.04
	20	41.69	41.76	41.98	0.03
	50	51.01	52.75	44.02	0.04
	90	38.46	38.46	40.71	0.03
	95	56.46	59.34	58.37	0.03
Over	0	60.51	53.85	51.56	0.03
	10	40.8	53.85	51.9	0.04
	20	58.13	57.14	49.07	0.03
	50	57.85	38.46	37.61	0.04
	90	57.78	47.25	53.68	0.03
	95	64.44	47.25	43.95	0.03

Table 5: Results for Label Propagation Classifier.

7.3 Significant Difference

Refer to *Appendix D* for more Friedman Tests of Significant Difference.

Resampling	Unlabelled %	F-Stat	p-value	Reject H_0?
None	0	3.818	0.148	False
	10	0.667	0.717	False
	20	1.273	0.529	False
	50	4.667	0.097	False
	90	3.818	0.148	False
	95	1.273	0.529	False
Undersampling	0	2.0	0.368	False
	10	0.545	0.761	False
	20	0.545	0.761	False
	50	1.636	0.441	False
	90	5.6	0.061	False
	95	3.714	0.156	False
Oversampling	0	2.0	0.368	False
	10	1.636	0.441	False
	20	3.818	0.148	False
	50	0.182	0.913	False
	90	2.0	0.368	False
	95	3.2	0.202	False

Table 6: Friedman Test for Each Combination of Resampling Method and Unlabelling Level (Averaging over Three Datasets).

F-Stat	p-value	Reject H_0?
7.238	0.027	True

Table 7: Friedman Test (Averaging over all Resampling Methods and Unlabelled Levels).

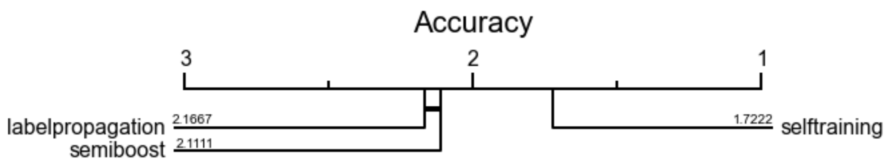


Figure 1: Nemenyi Critical Difference (Averaging over all Resampling Methods and Unlabelled Levels).

8 Discussion

8.1 Three Models over Three Datasets

Overall, I was able to train 162 separate models. Although my results were not always very good, I was still able to demonstrate the concepts learned during this project.

The Self Training algorithm seems to have performed well over all three datasets. Its worst results are in the *Heart* dataset, perhaps because this was the smallest dataset.Unlabelled entries could have lead to even less instances during training. This algorithm was very fast training.

The SemiBoost algorithm seemed to have performed better than the Self Training algorithm for the *Heart* dataset, but it performed atrociously bad on the *Online_Shopping_Intention* dataset. In this case, maybe the dataset was too large, noisy and unbalanced for this learner. In the smaller datasets or when under-/oversampling, this model seems to perform better. In addition, this was by far the slowest algorithm to train

Finally, the Label Propagation algorithm did moderately okay over all datasets. It did not outperform any of the other algorithms overall, but it did not perform very bad over any particular dataset either. This algorithm had a low execution time.

8.2 Unlabelled Levels

Primarily, the focus of this assignment was Semi-Supervised Learning, and comparing model predictions over six levels of unlabelled data. Please refer to Tables 3, 4, and 5 for the following discussion.

Unsurprisingly, all three classifiers achieved the highest F1-scores, accuracies, and AUCs for 0% unlabelled data. This did not depend on the dataset, nor the resampling method. With no unlabelled data, the models have more correctly labelled instances. Whereas with each increase of unlabelled data, we introduce the possibility of training error. An extreme example of this would if we unlabelled all *1*'s, so our models predicted *0*'s every time. In fact, this is possible in the highly skewed dataset *Online_Shopping_Intention*. More on the affect of class imbalanced later.

Evidently, as the amount of unlabelled data increased, the metrics of my three models decreased. Again, this was true for all datasets and resampling methods. However, there were some cases where the metrics would increase, as unlabelled data increased. This unexpected behaviour is most likely due to the different target distributions. Unlabelling was done randomly, meaning that each unlabelling level's dataset could inherit different information and/or noise from the original data. It is unclear why some models performed better for higher amounts of unlabelled data, but overall the trend still holds true-unlabelled data leads to worse predictions results. An evident example is the

Online_Shopping_Intention dataset and SemiBoost algorithm, where the F1-score for any level of unlabelling (other than 0%) and resampling was very low.

Additionally, the ROC curves (*Appendix A, B, C*) clearly show that only the 0% case consistently has good results and AUC. All other unlabelling levels fluctuate slightly above and slightly bellow the baseline. According to the 'No Free Lunch' theory no single model will be the best choice over all possible datasets. This is certainly true in my project, there each model performed dramatically different for all 54 cases (three datasets, three resampling methods, and six unlabelled levels).

8.3 Class Imbalance

I unable to see any improvement of results between skewed (no resampling), undersampled, and oversampled datasets. With some exceptions, it seemed that the prediction metrics were very similar between the three resampling methods, for each algorithm, unlabelled level, and dataset. As mentioned above, the 'No Free Lunch' theory tells us that no model always/never benefited from resampling of the data. Each case must be looked at separately- sometimes resampling methods let to improved accuracies, and sometimes they did not.

8.4 Comment on the Interplay Between SSL and Class Imbalance

Theoretically, the interplay of semi-supervised learning and class imbalance is very dangerous to classification. This was evident with when training the Semi-Boost algorithm over the *Online_Shopping_Intention* dataset. This dataset is highly skewed in favour of the negative class, so it is possible that during unlabelling we would remove more positive instances than negative. In reality, we are not usually removing labels, they are simply not present. This does not change that both unlabelled data and data skewness are complications to the classification problem, but these two things together magnify each other. Unlabelled data may lead to even more skewed data (ex. one label is harder to annotate), and resampling can lead to a higher proportion of unlabelled data (ex. oversampling un-/pseudo-labelled instances).

8.5 Analysis of Significant Difference

I performed several Friedman tests for significant difference of means. For each combination of resampling method and unlabelled data % (averaged over three datasets), I determined that the semi-supervised models were not significantly different (*Table 6*). For one particular combination, the methods are not significantly different. The methods are also not significantly different for different resampling methods (*Table 8*) or different unlabelling levels (*Table 9*). Thus, for any particular case, we would not choose one model over another.

Over all 54 cases, the Self Training model is significantly different from the other models (*Table 7*). The SemiBoost and Label Propagation algorithms are not significantly different from each other, and are significantly worse than the Self Training algorithms (*Figure 1*).

8.6 Best Overall Algorithm

The Self Training model is overall best, for the 54 possible cases. It has the highest F1-Scores, AUCs and accuracies, as well as the shortest execution times (*Tables 3, 4, 5*). It is has a significantly better accuracy, over all 54 combinations of dataset, resampling method, and unlabelled level (*Figure 1*).

Speaking to Explainable AI, interpretability, and trustworthiness, I would personally prefer to use the Self Training model. The algorithm is very easily explained, and one of the simplest semi-supervised classifiers available. The training and results can also be interpreted, since we can train any model (say a Decision Tree), visualize it, and finally display the pseudo-labels. The visualization can be done at any point in this iterative algorithm. Additionally, I would not have chosen to trust the SemiBoost algorithm I have adapted, since it is not part of a standard *sklearn* library, meaning it has not been peer-reviewed and is prone to bugs. The Label Propagation algorithm, although implemented by *sklearn*, is considerably harder to explain and visualize as it uses very large graphs.

9 Conclusion

In conclusion, I was able to implement and/or train three semi-supervised classifiers, namely Self Training, SemiBoost, and Label Propagation. I was able to discuss how the negative impact of unbalanced data is augmented in the semi-supervised setting.

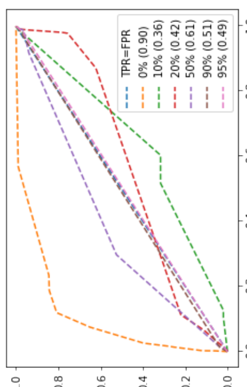
For specific resampling methods, or unlabelled levels, the three algorithms were not significantly different. However, overall the Self Training model was the best semi-supervised classifier, having the highest F1-score, accuracy (most of the time), and AUC, as well as the shortest execution time. The Friedman and Nenemyi tests of critical difference also confirmed that the Self Training model was significantly better than all others.

10 References

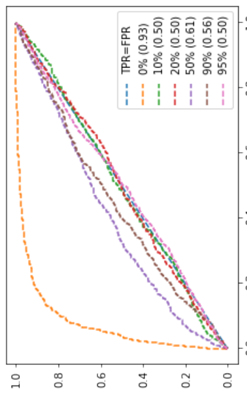
[1] Hassan Ismail Fawaz et al. “Deep learning for time series classification: a review”. In: *Data Mining and Knowledge Discovery* 33.4 (2019), pp. 917–963. URL: https://github.com/hfawaz/cd-diagram.

[2] Papabloblo. *SemiBoost*. June 2018. URL: https://github.com/papabloblo/semi_boost.

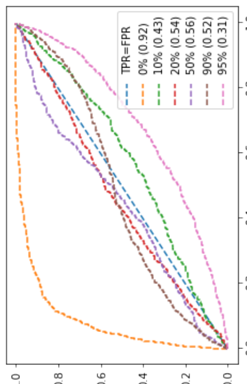
A Seft-Training ROC Curves



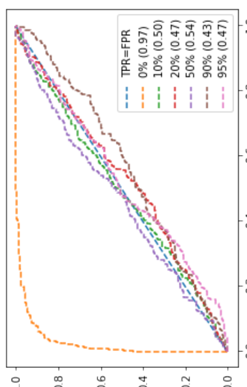
(a) Online_Shopping_Intention Skewed



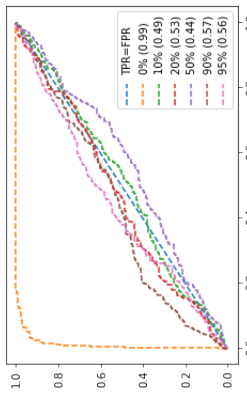
(b) Undersampled



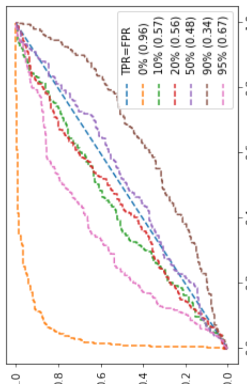
(c) Oversampled



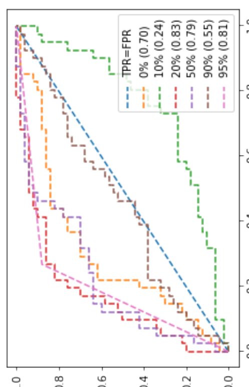
(d) Marketing_Campaign Skewed



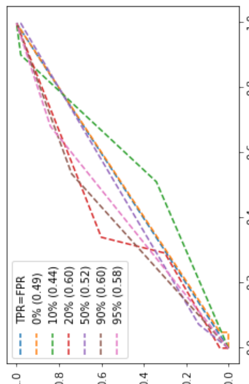
(e) Undersampled



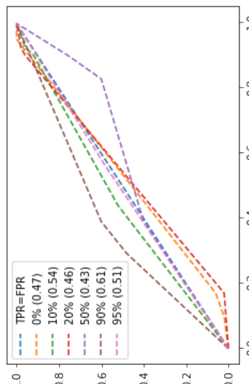
(f) Oversampled



(g) Heart Skewed



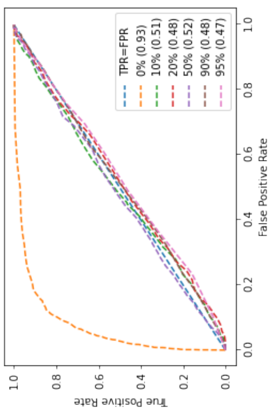
(h) Undersampled



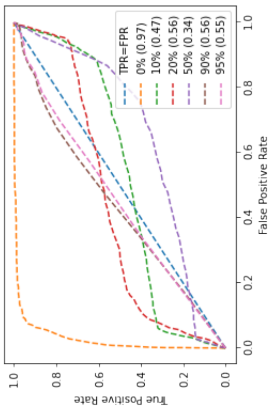
(i) Oversampled

Figure 2: ROC Curves for Self-Training Classifier, for Three Datasets With and Without Resampling.

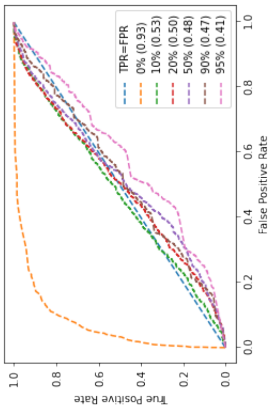
B SemiBoost ROC Curves



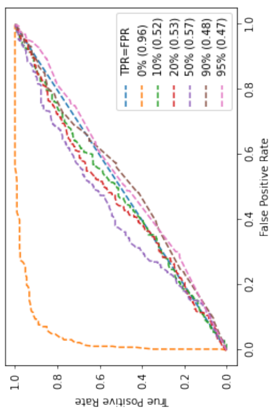
(a) Online_Shopping_Intention Skewed



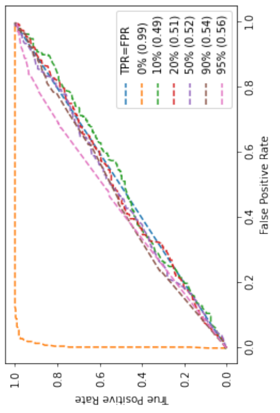
(b) Undersampled



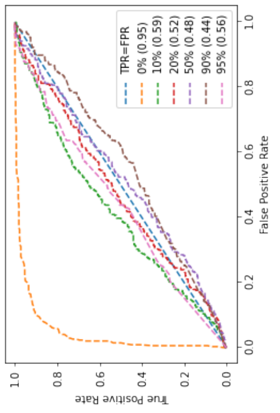
(c) Oversampled



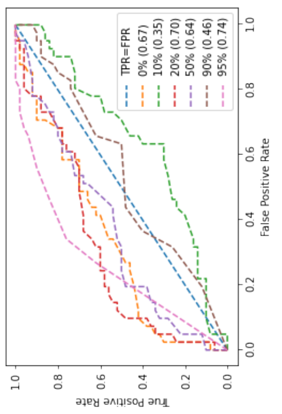
(d) Marketing_Campaign Skewed



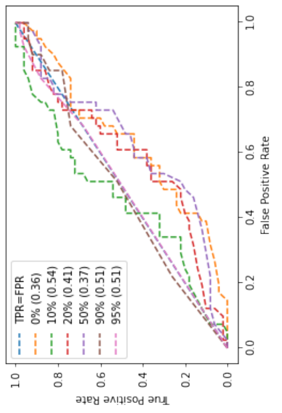
(e) Undersampled



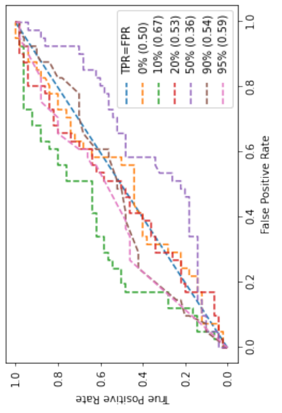
(f) Oversampled



(g) Heart Skewed



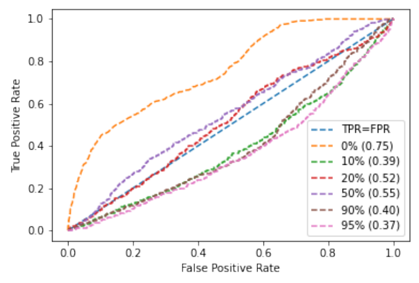
(h) Undersampled



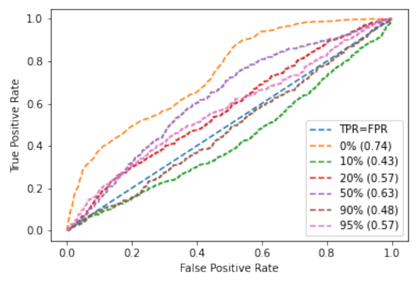
(i) Oversampled

Figure 3: ROC Curves for SemiBoost Classifier, for Three Datasets With and Without Resampling.

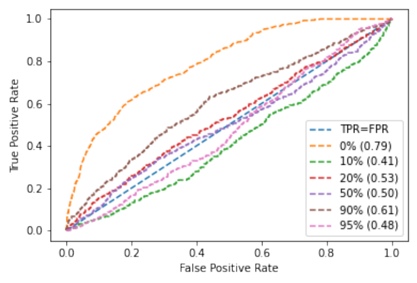
C LabelPropagation ROC Curves



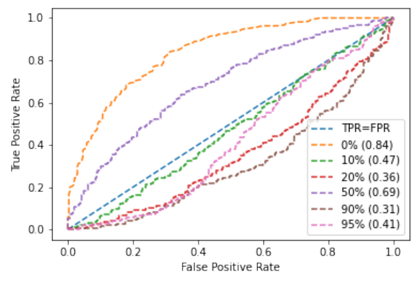
(a) Online_Shopping_Intention Skewed



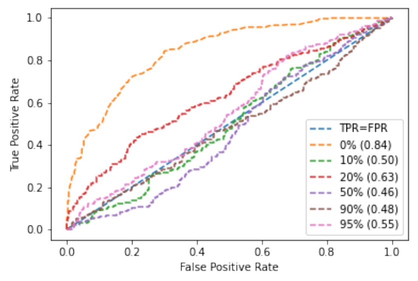
(b) Undersampled



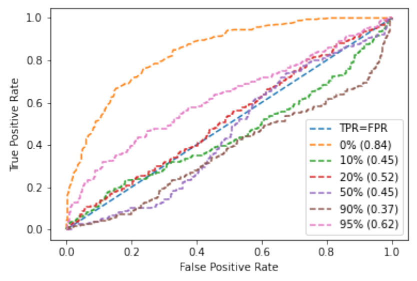
(c) Oversampled



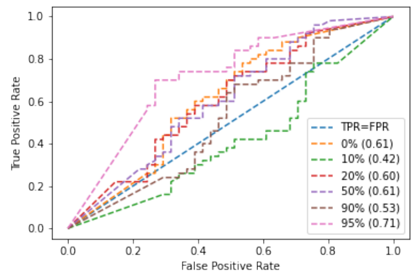
(d) Marketing_Campaign Skewed



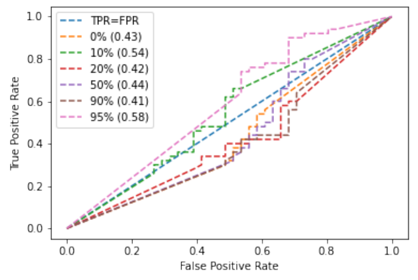
(e) Undersampled



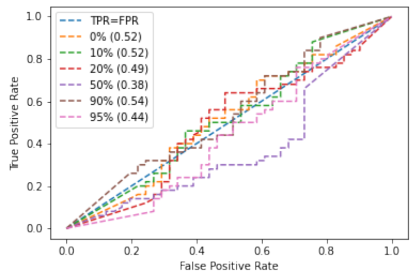
(f) Oversampled



(g) Heart Skewed



(h) Undersampled



(i) Oversampled

Figure 4: ROC Curves for Label Propagation Classifier, for Three Datasets With and Without Resampling.

D More Friedman Tests

Resampling	F-Stat	p-value	Reject H_0?
None	4.667	0.097	False
Undersampling	4.667	0.097	False
Oversampling	4.667	0.097	False

Table 8: Friedman Test for Each Resampling Method (Averaging Unlabelled Levels).

Unlabelled %	F-Stat	p-value	Reject H_0?
0	0.667	0.717	False
10	0.667	0.717	False
20	4.667	0.097	False
50	0.667	0.717	False
90	0.545	0.761	False
95	2.667	0.264	False

Table 9: Friedman Test for Each Unlabelled Level (Averaging Resampling Methods).