Homework 3 – 19.05.2021

Oğuz Bulut Kök

I have used three supervised algorithms to predict the voter turnout. As I thought that each variable might have a merit on its own, I kept all of the variables (even though it might risk a sinkhole model/overfitting). Before doing my run, I prepared my data by adding all of the NaN values (they were coded differently in the initial data), replacing them with mean values and converting Boolean dependent variable to integers. Mainly I used random train test splits for a simple method to train my Machine Learning algorithms. To reduce the time of processing, I have also used ANOVA-F values to select some of the features[1] and normalizer to normalize my data. Additionally, I used methods of inverse weighting and oversampling to overcome inequality of predictors[2]. Finally, I used accuracy scores and classification table for random forest for computing the accuracy of the model, confusion matrices for visualising accuracy/fit and cross validation scores to also measure the fit.

For my first algorithm, which is Support Vector Machine, I expected it to perform poorly due to large data/features. Therefore, I was not surprised with the initial results where it had subpar accuracy/high overfitting. I tried to weigh the SVM in the second one, which reduced accuracy but provided a more diverse distribution and the highest cross validation score, which makes it more suitable for other datasets. In both of the models, there were high numbers of false positives and negatives, especially in weighted one (see confusion matrix).

Secondly I tried to use logistic regression as my second machine learning algorithm which gave the same results with SVM (without weights). And the results remained unchanged when I ran it with weights. Even though its "accuracy" was high in the both of the models, clustering over true positives in confusion matrices suggests that it is not-so reliable as a model for training a machine.

Thirdly, I tried a more complex algorithm, random forest. When I first ran it, just like others, it performed poorly in predicting 0s (not included in the code). However, when I oversampled 0s, it became highly accurate to represent the model while having similar fit with other algorithms. (see cross validation score).

Shortly, the main problem between these models is the distribution of predictions, as seen in the confusion matrices. There seems to be some underfitting/overfitting due to simple

[1] I also ran my data without feature selection which did not change the results drastically.
[2] 1s were 4.6 times higher than 0s.

training method that I have used and highly asymmetric dependent variable. Logarithmic and normal SVM models are highly underfit for predicting 0 while weighted SVM and random forest perform better. As weighted SVM gives higher amount of false positives/negatives, the cross-validation scores are rather close between two models, and the random forest performs better accuracy-wise; therefore, I chose random forest as my main model.

Figures

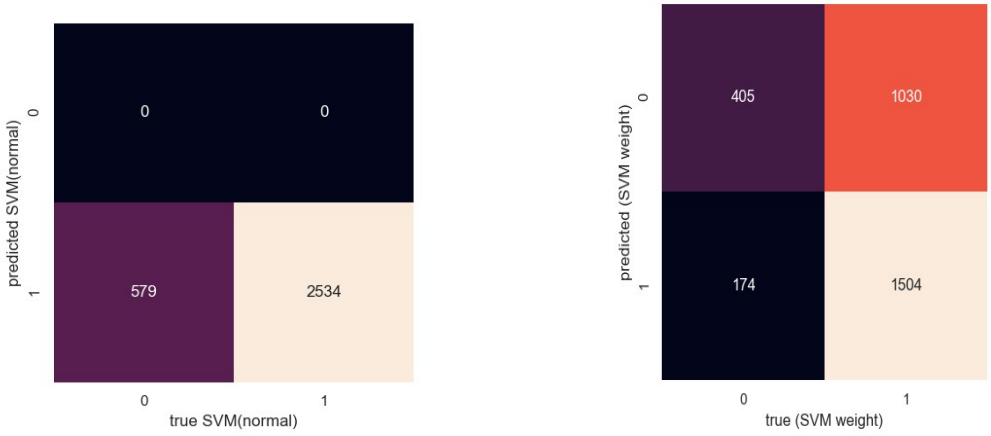

Figure 1. Confusion Matrices for SVM models

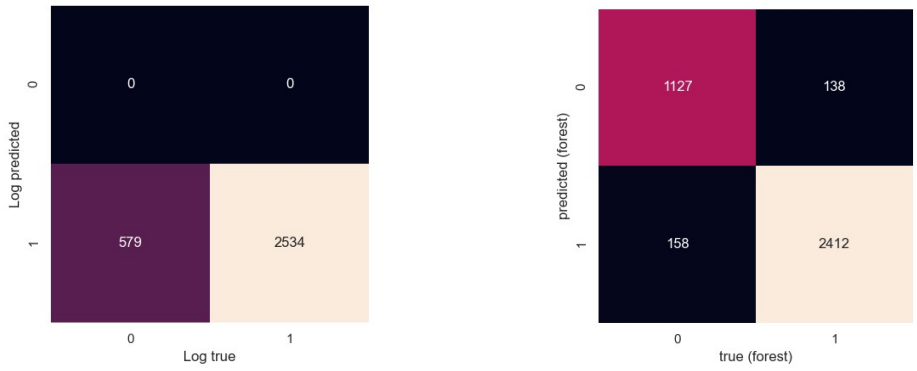

Figure 2. Logistic Regression Confusion Matrix / Random Forest Confusion Matrix

Table 1. Random Forest Classification Report

	Precision	recall	F1-score	Support
0	0.90	0.89	0.89	1301
1	0.94	0.94	0.94	2534
accuracy			0.92	3835
Macro avg.	0.92	0.91	0.92	3835
Weighted avg.	0.93	0.93	0.93	3835

Table 2. Accuracy and Cross-Validation Scores for all of the algorithms

	SVM	SVM(weighted)	Logistic	Logistic (weighted)	Random Forest
Accuracy	0.814	0.574	0.814	0.814	0.928
Cross-Validation Score	0.651	0.705	0.655	0.655	0.674