Machine Learning and Political Ideology in the U.S.: An Experimental Case Study using SMOTE

Héctor Bahamonde [*1] and

Cristobal Quininao [†2]

[1]Assistant Professor, Instituto de Ciencias Sociales, O'Higgins University

[2]Assistant Professor, Instituto de Ciencias de la Ingeniería O'Higgins University

April 14, 2021

[*]hector.bahamonde@uoh.cl; www.HectorBahamonde.com.

[†]cristobal.quininao@uoh.cl; https://cquininao.wordpress.com.

Authors are listed in alphabetical order. This project was funded by the Center for [ ] at [ ].

Abstract

Abstract here.

Abstract length: 2 words.

Please consider downloading the last version of the paper here.

Keywords— United States; machine learning; SMOTE; ideology.

I. Ideology

II. Machine Learning in Political Science

"inductive learning" Cranmer2019, p. 1

"we use ML for prediction" Cranmer2019, p. 1

"ML algorithms can usually predict outcomes with greater accuracy than the standard regression-type models we use in political science (often *much* greater accuracy)"[1] Cranmer2019, p. 1

"95 percent and 100 percent accuracy when used to construct a classification tree using the entire data set." Schrodt1990, p 52

"introduce a novel method to diagnose electoral irregularities using vote counts. Specifically, we propose the use of machine learning techniques to detect fraud" Cantu2011a p 401

III. Data

IV. Materials and Methods

The overall model training and validation process consists in a three phases. First the database processed by eliminating repeated observations. Then the data are normalized employing the MinMax scaler. Exploiting the normalized dataset, two different experiments were performed. In the first experiment machine learning algorithms were executed to classify survey participants according to their ideological preferences in a liberal-conservative scale. For classification purposes, the ideological preferences of survey participants were recoded in weather they were liberals or conservatives. Since the resulting training pairs datasets are strongly imbalanced, prior to the construction of the learning

las repetidas son por el diseno con-joint?

citar: que es, de donde sale, que hace

hb: de cuanto era la escala

era "very" o todo lo que era hacia lib-

[1] Emphasis in original.

models implemented in this paper, the SMOTE oversampling procedure was used (Chawla et al. 2002). Five nearest neighbors for each vote-selling sample were employed (Artetxe et al. 2020). The resulting training data pairs were then validated via a repeated cross-validation process (RCV). The RCV process consists of constructing several machine learning models by using the same dataset but under the idea of generating folds and repetition. In particular, each cross-validation repetition citar consists of partitioning the dataset in a particular number of folds, using each fold as a test dataset while using all the remaining data for model training. Finally, average performance measures across all repetitions were reported. In this manuscript, the reported results are the average of the 30 repetitions of the RCV models.

I. Classification Methods

Within the predictive model framework, several machine learning approaches can be used for analyzing the data (Vapnik 2013; Witten and Frank 2002; Maimon and Rokach 2005). Due to the small dataset used in these experiments, applying deep learning algorithms were discarded (Balas et al. 2019). Therefore, we focus in the following classical well-known methods: *Support Vector Machines*, *Multilayer Perceptron* and *Naïve Bayes method*.

Support Vector Machines (SVM) Support vector machines (SVMs) are a set of related methods for supervised learning which have applications to classification (the output data is a sequence of tags) and regression (the output data is a continuous variable) problems (Maimon and Rokach 2005, Ch. 12). When using SVM for classifying purposes, the goal is finding linear functions that are able to separate the data based on the support vectors—or "boundaries"—of the different classes. The idea of the algorithm is to find the optimal hyperplane, i.e., the hyperplane that maximizes the distance between the boundaries of each class. This can be done by implementing a

quadratic programming problem (Vapnik 2013). When the classes are not linearly separable, then it is possible to place the data into a space of higher dimensionality using a kernel trick (Vapnik 2013; Maimon and Rokach 2005) so that the transformed dataset become linearly separable. We used the SVM library *Sklearn* implemented in `Python` for training and estimation purposes. Both the linear and Radial Bases Functions (RBF) were implemented.

"trick" termino tecnico? o hace ref a via alternativa?

Multilayer Perceptron Multilayer perceptron (MLP) is the classical feed-forward artificial neural networks (ANN) composed by multiple densely interconnected layers of computational units, usually known as "artificial neurons" (Wolff et al. 2019). The method corresponds to the family of supervised learning algorithms that find a function able to map multidimensional samples. In this application the focus is on mapping the socio-demographic battery of questions administered to survey participants into the vote-selling question. MLPs are constructed as follows: (1) a number of hidden layers is fixed , (2) the architecture starts by defining m artificial neurons—i.e. the set of features of the input data corresponding to the input layer, (3) any artificial neuron in the first hidden layer integrates the inputs from the input layer and combines the input values with a weighted linear summation (the weights become part of the parameters to be tuned in the learning phase), (iv) the result of this summation is nonlinearly transformed through an activation function (for instance an hyperbolic tanh function). The procedure is repeated for each hidden layer, until the output layer is reached. The connection weights can be learned from data applying the back-propagation algorithm (Haykin and Network 2004). The training and estimation processes were executed via the *MLPClassifier* module implemented in the *Sklearn* `Python` library. The *MLPClassifier* implements a MLP algorithm that trains using back-propagation through a stochastic gradient descent. The rectified linear unit function for the activation of the hidden layers was used. The MLP model constructed has two hidden layers, each of 50 neurons.

meaning de hidden? fixed?

architecture?

me perdi a cagar!

activation?

Naïve Bayes Method Naïve Bayes methods are a set of supervised learning algorithms that apply Bayes' theorem with the "naive" assumption that all features of an individual are independent. Bayes' theorem provides a rule to calculate the conditional probability of an event given some prior knowledge. This method allows estimating the joint probability distribution of a feature vector as the product of the unidimensional distribution probabilities of each feature (Wolff et al. 2019). In this application the *GaussianNB* module from *Sklearn* `Python` library implements the Gaussian Naive Bayes algorithm. The likelihood of the features is assumed to be Gaussian.

II. Classification Performance Metrics

We illustrate the chosen sensitivity metrics with the liberal classification machines only—the conservative situation is identical. At the end of each cross-validation fold, two different vectors are produced: y_{test} corresponding to the actual responses given by survey participants to the liberal question (restricted to the fold used for testing), and y_{pred} corresponding to the predicted answers given by the model for the input features. With these two vectors at hand it is possible to compute the following quantities:

decia straightforward

- True positive counts (TP): number of survey participants such that the model predicts as very liberal and they actually belong to the very liberal class.
- True negative counts (TN): number of survey participants such that the model predicts as not very liberal and they do not belong to the very liberal class.
- False positive counts (FP): number of survey participants such that the model predicts as very liberal but they do not belong to the very liberal class.
- False negative counts (FN): number of survey participants such that the model predicts as not

very liberal but they actually belong to the very liberal class.

With these numbers, the recall (R)—or model sensitivity—was computed.[2] Particularly, the positive predictive value (PPV) or "precision"[3] as well as the f-score were computed considering the following relationships,[4]

$$R = \frac{TP}{TP + FN}, \quad PPV = \frac{TP}{TP + FP}, \quad f = \frac{2}{\frac{1}{R} + \frac{1}{PPV}}. \tag{1}$$

The average and the standard deviation across all cross-validation folds are also reported in Table 2.

An alternative metric is the accuracy metric (A) defined as,

$$A = \frac{TP + TN}{TP + TN + FP + FN} = \frac{TP + TN}{\text{size of the fold}}. \tag{2}$$

However, considering that the data are strongly imbalanced, metrics defined in Equation 1 and Equation 2 are more informative. Another metric widely used to compare binary classifiers is the Receiver Operating Curve (ROC) (Wolff et al. 2019; Artetxe et al. 2020) which plots sensitivity R defined in Equation 1 against the false positive rate (FPR) defined as follows,

$$FPR = \frac{FP}{FP + TN}. \tag{3}$$

The ROC metric is computed as a probability curve, and the area under the ROC curve represents the degree of separability. In other words the ROC quantifies how much the model is capable of distinguishing between classes. In this application the higher the area under the curve, the better

[2] The fraction of examples classified as very liberal, among the total number of very liberal examples.
[3] The fraction of true very liberal examples among the examples that the model classified as very liberal.
[4] A perfect model has an f-score of 1.

the model is at predicting liberals as liberals and non liberals as non liberals. Remark that (1) a perfect model has an area under curve of 1, (2) a model with an area under the curve of 0 implies that the model is actually reciprocating the classes, therefore simply relabeling is enough to get good results; and (3) the worst-case scenario is when the area under the curve equals 0.5, meaning that the model has no class separation capacity.

To actually compute these metrics every fold test is compared against the remaining folds as training dataset. Since the resulting training data set is imbalanced, the SMOTE oversampling technique with five neighbors on the minority class is used. Remark that the test set remains imbalanced which has some consequences. In particular, even one misclassified sample translates into large reductions of the performance measures. Moreover, the number of remaining liberal samples in each fold depends on the number of folds, thus having consequences in the imbalance ratio. We report results with decreasing number of RVC folds in order to test the stability of the results.

compared,

ta bn?

here

V. Results

Tables 1 and 2 show the results for metrics mentioned above of the machine learning techniques after 40 repetitions (3 fold), 10 repetitions (4 fold), 8 repetitions (5 fold) and 4 repetitions (10 fold) of the RCV experiments with SMOTE class imbalance correction. There is no statistical evidence to argument that the number of folds has an effect on the performance of each method. An F- test over the number of folds shows that there is no statistically significant difference $p \gg 0.1$.

If we test the performance of each methods we find some differences, specially on the PPV, f-Score and AUC metrics. SVM with RBF kernel is above SVM Linear, MLP, and NB independently of the number of folds. On the other hand, the recall metric (R), shown in Table 1 (left), shows no sufficient evidence to affirm that methods perform differently. Indeed, applying an F-test over the

different type of machines for the each folds number we find no strong statistical evidence, and it would depend on the confidence level whether or not we reject the null hypothesis (for instance, the p-value es approx 0.026 for 10 folds). However, the result changes dramatically, if we discard the MLP model which performs below any other model ($p \approx 0.77$ for 10 folds). We conclude that according to the recall metric, all methods except for MLP have similar performance. Figure 2 shows the ROC curves for all approaches in the case of RCV with 5 folders.

The f-scores shown in Table 2 confirm that SVM with RBF kernel improves over SVM Linear, MLP and NB regardless of RCV number of folders. An F-test carried out over these results confirms that the performance differences between predictive models are statistically significant ($p \approx 0.002$) . Specific one-sided t-tests comparing each pair of modeling approaches confirms that SVM with RBF kernel perform better than SVM Linear, MLP and NB. However, the superiority of SVM with RBF kernel relative to MLP is less pronounced ($p \approx 0.05$). On the contrary, the effect is more pronounced for the AUC metric. The F-test carried out over these results confirms ($p \ll 0.001$), t-test pairwise tests ($p < 0.001$).

Table 1: *Average ± standard deviation Recall R (left) and positive predictive value PPV (right) performance of SVM (linear), SVM (RBF), MLP and NB for decreasing number of folders in the repeated cross validation process. All results are calculated with SMOTE oversampling correction of class imbalance.*

		SMOTE		
nfolds	SVM (linear)	SVM (RBF)	MLP	NB
10	0.68±0.07	0.67±0.08	0.65±0.07	0.68±0.08
5	0.68±0.05	0.67±0.04	0.65±0.06	0.66±0.06
4	0.67±0.03	0.66±0.04	0.64±0.04	0.67±0.04
3	0.68±0.03	0.67±0.05	0.67±0.05	0.67±0.05

		SMOTE		
nfolds	SVM (linear)	SVM (RBF)	MLP	NB
10	0.45±0.06	0.52±0.08	0.47±0.07	0.44±0.07
5	0.46±0.05	0.52±0.07	0.48±0.05	0.44±0.05
4	0.45±0.04	0.52±0.06	0.47±0.04	0.43±0.05
3	0.45±0.03	0.52±0.04	0.45±0.03	0.44±0.04

Table 2: *Average ± standard deviation f-score (left) and AUC (right) performance of SVM (linear), SVM (RBF), MLP and NB for decreasing number of folders in the repeated cross validation process. All results are calculated with SMOTE oversampling correction of class imbalance.*

		SMOTE		
nfolds	SVM (linear)	SVM (RBF)	MLP	NB
10	0.54±0.06	0.58±0.07	0.54±0.06	0.53±0.06
5	0.55±0.05	0.59±0.05	0.55±0.05	0.53±0.05
4	0.54±0.03	0.58±0.04	0.54±0.03	0.52±0.04
3	0.54±0.02	0.58±0.03	0.54±0.03	0.53±0.04

		SMOTE		
nfolds	SVM (linear)	SVM (RBF)	MLP	NB
10	0.68±0.04	0.72±0.05	0.68±0.04	0.67±0.04
5	0.67±0.03	0.72±0.03	0.68±0.03	0.67±0.03
4	0.67±0.02	0.72±0.04	0.67±0.03	0.67±0.03
3	0.68±0.02	0.71±0.02	0.68±0.02	0.67±0.02

References

Artetxe, Arkaitz et al. (2020). "Balanced training of a hybrid ensemble method for imbalanced datasets: a case of emergency department readmission prediction." In: *Neural Computing and Applications* 32.10, pp. 5735–5744.

Balas, Valentina Emilia et al. (2019). *Handbook of deep learning applications*. Vol. 136. Springer.

Chawla, Nitesh V et al. (2002). "SMOTE: synthetic minority over-sampling technique." In: *Journal of artificial intelligence research* 16, pp. 321–357.

Haykin, Simon and N Network (2004). "A comprehensive foundation." In: *Neural networks* 2.2004, p. 41.

Maimon, Oded and Lior Rokach (2005). "Data mining and knowledge discovery handbook." In:

Vapnik, Vladimir (2013). *The nature of statistical learning theory*. Springer science & business media.

Witten, Ian H and Eibe Frank (2002). "Data mining: practical machine learning tools and techniques with Java implementations." In: *Acm Sigmod Record* 31.1, pp. 76–77.

Wolff, Patricio et al. (2019). "Machine learning readmission risk modeling: a pediatric case study." In: *BioMed research international* 2019.

Word count: 2,018

VI. Appendix

I. Appendix 1