EE 565 - Machine Learning I
Project 6 Report

Jorge A. Garcia

***Abstract*—In this report, the use of a Decsion Tree for classification is explored using the Double Moon data set. Prediction of the 2018 New Mexico state representative elections is done using a Decision Tree, as an example of a real-world problem for future use in the course.**

***Index Terms*—Machine Learning, Data Science**

I. Introduction

The last algorithm to be explored in this course is the Decision Tree. By measuring the attributes in a data set to a set of rules, the data points are separated into different segments until a node consists of as many instances of a single class type, considering the majority the predicted label for that route.

The real-world problem considered in this project is that of predicting House of Representative elections for 2018 in New Mexico, using a mix of political opinion survey and census survey data sets to achieve this. Given the nominal nature of the data, a Decision Tree is also used to approach this problem.

II. The Decision Trees

The double moon distribution is used to generate two different data sets for this section, with parameters $r = 1$, $w = 0.6$ and $N = 3000$ set for both. They separation between the upper and lower moons varies, with $d = 0.5$ and $d = -0.5$ each.

A. Library Used: Scitkit-learn

For this report, the *DecisionTreeClassifier* class provided in the *scikit-learn* library [1] for Python is used. This is a very robust implementation, with a variety of options for the user to consider. Amongst the most relevant are the choice between Gini and Entropy splitting, the maximum depth of the tree, the minimum number of samples to split at a node, the maximum number of leaf nodes allowed and the minimum impurity at which to split. For this section, Entropy is used to determine node splitting, and no constraints are set for splitting or tree size.

B. Classification of Double Moon distribution

After model training, the resulting tree for the data set with $d = 0.5$ is shown in Figure 1. The tree is 1 split deep with 2 leaf nodes, being able to perfectly split the data set with a single decision. This makes sense, as this is a linearly separable problem, as seen in Figure 3.

The decision tree for the data set using $d = -0.5$ is seen in Figure 2. This tree ended up being 5 splits deep, with a total of 6 leaf nodes to predict the data. The model is also able to separate this data set, as with enough lines one can draw a decision boundary that splits both classes perfectly as seen in Figure 3.

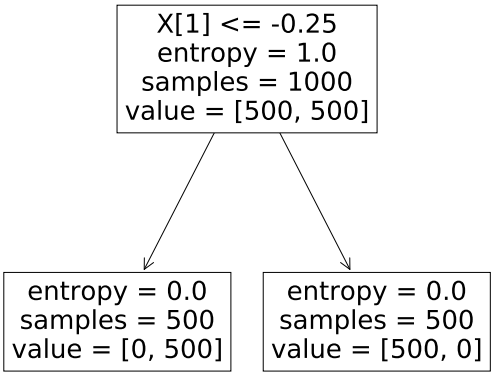

Fig. 1. Decision tree for the Double Moon distribution with $d = 0.5$.

III. Prediction of 2018 New Mexico State Representative Elections

The data set used in this section is one that I compiled and created during a 2018 summer research experience at the Joint Institute for Computational Sciences, which was used to experiment with predicting state representative elections using machine learning algorithms supposing only two parties. It is actually composed of two different data sets that are altered such that they are formatted in the same manner, allowing them to be merged. The training set is from political opinion surveys from 1992 to 2012 from the American National Election Survey [2], consisting of 5526 entries. The testing set is then a state's census survey data provided by the U.S. Census Bureau [3], filtered to only include respondents who are eligible voters. At the time the data set was created, 2016 census data from Texas, California, Florida and Minnesota were used as different testing sets. I updated the testing data set to be 2018 census data from New Mexico to make the problem

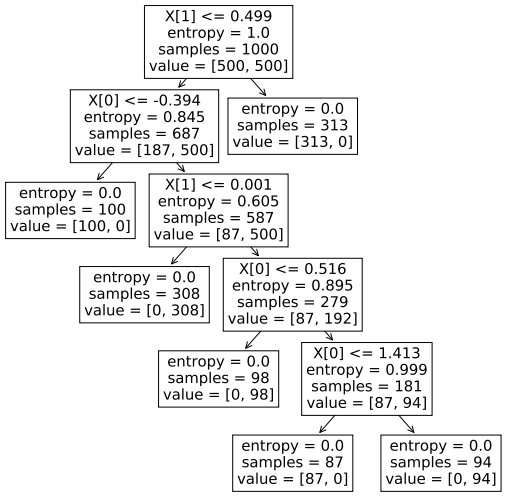

Fig. 2. Decision tree for the Double Moon distribution with $d = -0.5$.

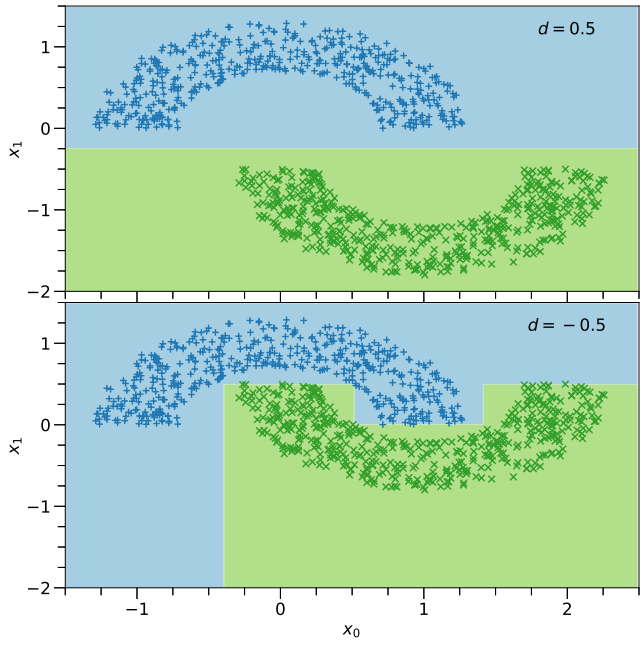

Fig. 3. Prediction of training data set class, as well as decision region for both double moon distributions with $d = 0.5$ (top) and $d = -0.5$ (bottom).

more familiar, with a total of 10636 entries. The attributes considered are a voter's age, gender, race/ethnicity, census region, income group, occupation, work status, education and marital status. The corresponding output for each entry in the training set is the party voted for (0 for Democrat, 1 for Republican). Further details about the attributes can be found in the "election_data_info.txt" file.

The goal of the student is to train a model that is able to predict the results of the 2018 congressman elections for New Mexico using the provided data. Since the all of the considered attributes are nominal data, a Decision Tree classifier using the *Scikit-learn* library in Python was chosen to approach this problem as it can be quickly implemented and one-hot-encoding of the attributes is unnecessary. One-hot-encoding of all the attributes would transform this from a 9-dimensional data set into a 47-dimensional sparse data set, which may cause issues when using other algorithms.

First we must find the best model to approach the given problem. The hyperparameters to optimize are criterion to measure node impurity (Entropy or Gini index), and maximum tree depth. Evaluation of a model's performance is done by considering 10-fold cross validation, that is 10 variations of 4973 data points for model training and 553 for validation. The average training and validation accuracy for each trained model is measured across the 10 folds, varying tree depth from 1 to 150 splits for both Entropy and Gini index. The resulting prediction accuracy for the validation set can be seen in Figure 4. Use of Entropy promotes higher accuracy with fewer splits, with a tree depth of 24 splits achieving the maximum accuracy possible of $84.38 \pm 1.30\%$ for the validation set. Considering a Democratic vote a negative case (0) and a Republican vote a positive case (1) [1], these settings achieve a true positive rate of $80.48 \pm 1.14\%$ and a true negative rate of $87.5 \pm 2.09\%$, showing some model bias towards predicting for Democrat votes. These hyperparameter settings are then the ones considered for the rest of the section.

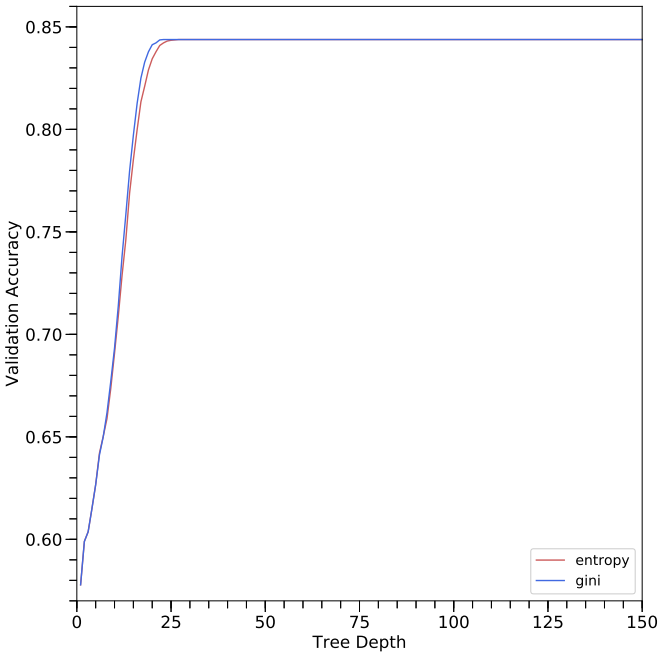

Fig. 4. Accuracy of the validation set as tree depth increases, using Entropy (red) and Gini index (blue) criteria.

The state census data contains no information on the party voted for, so evaluation of the testing set is done by considering the overall election result predicted rather than a prediction

[1]Not trying to be political here, it's just how the data is formatted.

accuracy for each entry. Each test entry contains a weight w_i that represents how many votes a particular respondent is worth. This weight takes into account the number of people a respondent represents in the state population, as well as voter turnout within the state due to race [4]. The total number of votes in the election is then $N_{\text{tot}} = \sum_i w_i$, and similarly the votes for a party are considered with the subset of weights with matching predicted labels.

Prediction of the census survey data is done by again training 10 classifiers using the 10 folds from the training set using the Entropy criterion and a tree depth of 24 splits. This is to obtain an uncertainty in the prediction of election results, as there will be slight variations from what the model is trained on. After prediction and tallying votes, Democrats were predicted to receive $58.78 \pm 3.14\%$ of the votes in the state in 2018, with Republicans receiving the remaining 41.22 ± 3.14 of votes. For comparison, the actual results for the 2018 House of Representative elections in the state were 58.27% for the Democratic party and 38.18% for the Republican party [5]. Given that the model provides an accuracy prediction within the uncertainty, this demonstrates that demographic characteristics are strong indicators of who people are likely to vote for.

IV. Conclusion

In this report, we experimented with the implementation of a Decision Tree classifier on the Double Moon distribution, effectively being able to separate both classes given enough splits. The Decision Tree was then put to the task of predicting the results for the 2018 New Mexico congressman elections, with the obtained prediction being in agreement with the actual election results.

References

[1] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[2] American National Election Studies, "ANES Time Series Study." [Online]. Available: https://electionstudies.org/data-center/

[3] United States Census Bureau, "American Community Survey Public Use Microdata Sample." [Online]. Available: https://www.census.gov/programs-surveys/acs/data/pums.html

[4] ——, "Voting and Registration in the Election of November 2018." [Online]. Available: https://www.census.gov/data/tables/time-series/demo/voting-and-registration/p20-583.html

[5] C. L. Johnson, "Statistics of the Congressional Election of November 6, 2018," 2019. [Online]. Available: https://history.house.gov/Institution/Election-Statistics/Election-Statistics/