Daniel Sullivan

Home Work 2

Pre-work

1. Read this blog: https://decizone.com/blog/the-good-the-bad-the-ugly-of-using-decision-trees which shows some of the issues with decision trees
2. Choose a dataset from a source in Assignment #1, or another dataset of your choice.

Assignment work

1. Based on the latest topics presented, choose a dataset of your choice and create a Decision Tree where you can solve a classification or regression problem and predict the outcome of a particular feature or detail of the data used.
2. Switch variables to generate 2 decision trees and compare the results. Create a random forest for regression and analyze the results.
3. Based on real cases where desicion trees went wrong, and 'the bad & ugly' aspects of decision trees (https://decizone.com/blog/the-good-the-bad-the-ugly-of-using-decision-trees), how can you change this perception when using the decision tree you created to solve a real problem?

Deliverable

1. Essay (minimum 500 word document)
   Write a short essay explaining your analysis, and how you would address the concerns in the blog (listed in pre-work)
2. Exploratory Analysis using R or Python (submit code + errors + analysis as notebook or copy/paste to document)

The dataset I chose from homework one was the pokedex data set I found on Kaggle here: Pokemon Stats | Kaggle. It contains the stats for every Pokémon in all games as well as what generation they were made as well as if they were legendary or not and their type. For this homework I decided to do a classification on whether or not a Pokémon is legendary or not. To give background legendary Pokémon are typically very powerful and are for the most part associated with creating parts of the world or being almost deities in their universe and there is only one of each legendary Pokémon. I made four different attempts to model this classification. I made two decision trees one with all variables and another where the variables were restricted. Additionally I made a random forest and a adaboost model to see how these ensemble methods work.

Decision Trees:

I started by including all of the variables together and made the decision tree using the ctree function. Looking at the plot I saw the tree contained three nodes the first separated by whether the total stats were over or under 579. The next two nodes separated by generation selecting generation 5 as a cutoff if total stats were under 579 and generation 3 if over 576. I found this very interesting since generation I don't think should have a strong correlation with any of this and am suspicious this may be an aspect of overfitting. To look at this I wanted to restrict out the generation variable and chose to move forward with three variables with the highest correlation to the legendary Pokémon, total stats, attack and sp attack. This restricted variable tree interestingly enough made just a single node, whether the total stats are over or under 579. Surprisingly this improved the test data classification. For the first model there was an accuracy of 0.937, sensitivity of 0.984, and specificity of 0.5405. The second model

with the single node hade an accuracy of 0.947, Sensitivity of 0.965, and specificity of 0.81. I think with this data in mind the first decision tree may have over fit the training data.

Ensemble methods:

The two Ensemble methods unfortunately did not improve upon the decision trees too much. For the random forest with all variables I got an accuracy of 0.0.949, Sensitivity of 0.985, and specificity of 0.676. The adaboost model had an accuracy of 0.953, Sensitivity of 0.844, and specificity of 0.965. Unfortunately for these approaches it is much harder to check under the hood to identify nodes and where each model determines the limits for making the classification.

Conclusions:

It is clear the two top winners of the four are the adaboost model had an accuracy of 0.953, Sensitivity of 0.844, and specificity of 0.965 and the restricted decision tree with accuracy of 0.947, Sensitivity of 0.965, and specificity of 0.81. with the adaboost model being slightly better but not significantly. While this may seem like very good measures I am a bit skeptical of how useful these models would be on an extensive testing dataset. I believe that there is not a great amount of efficiency in the models due to the class we are trying to predict only accounting for 10% of our entries. this makes the model considerably harder to train from the dataset. Especially for bootstrapping methods like the random forest on an already small dataset. The dataset that should deal with this aspect the best would be Adaboost as it is built in a way that would be sensitive to outliers which is most of the “legendary” class.

As far as the article stating the good the bad and the ugly on decision trees I found that my niche case with this data didn’t really fit the picture that was being painted. Yes the data set I chose when interpreted with decision tree models was very easy to interpret and the implementation was straight forward and gave a simple solution, however I did run into some issues that I did not expect. For starters I am fairly certain I saw some over fitting with my first all variable model where decision trees are supposed to be difficult to overfit. When I am faced with changing and reflecting on how I can change the perception of the bad aspects and downfalls of decision trees I think there are two main ways to tackle this. First to tackle their issues with accuracy and over generalizing at times this can be tackled by implementing many of the ensemble aspects, trading off some of the good(interpretability, intuitiveness, ease of optimization) in order to make a better predictive model. This would be in the form of boosting with gradient or adaboost or bagging with random forests. The second would be to understand that decision trees are one tool in a toolbox. There are limitations to its application and points where they can be optimal and other places where other algorithms shine.