**APM Project Outline (10/16/2017)**

**March Madness Predictions:**

Group Members: Ryan Conklin (rc45852), Bryan Lanier (bgl375), Chris Lunger (csl946), Raksha Pai (rp33426), Siddhant Shah(sss3976)

Background:

For our project we will be using historical data in order to try to predict the results of the 2017 March Madness college basketball tournament. Every year millions of people make brackets in order to compete against their peers and colleagues to see who is best at predicting results. There are 68 teams competing and 67 games played, and the propensity of ‘upsets’ draw many towards getting invested in the drama of the tournament. There has never been an instance of someone filling out a perfect bracket; we hope to be the first.

The Data:

We will be using the 2017 March Madness Dataset from the Kaggle competition that occurred earlier this year1. They have included a large variety of data, including data about every single regular season game, including teams, location, time, and overall team stats such as shots, points, personal fouls etc. It also includes tournament data, such as seeds and regions of each team, which is important for the tournament structure itself.

As for additional data, there is similarly structured data from their previous competitions going back to 2014 which we may use to add even further historical trends. There is also years and years of historical tournament data which we could use to try to figure out trends or idiosyncrasies of the tournament, however this may turn out to be a lot of noise. Scraping player level data for the season for each team is also a viable option that could be implemented. Some teams may have a winning record over other teams, and that may be useful to take into account if we are to predict potential upsets.

This is ultimately a many-step classification problem, and there are many methods we can use to approach it.

The Method:

To predict the entire bracket with the best accuracy, we would like to predict the winner of each match in the bracket. Even though the winners advance on to the next round, we chose to represent the probability of winning games as a markov process, where the probability of winning each game is independent of all previous games2. Predicting the correct outcome and standing for every team is very difficult – one upset during the actual tournament, especially if it’s early, could make our predicted bracket very inaccurate. Thus, we would like our model to have a high confidence with respect to predicting winners.

Before we get into which models to use, a little bit more about the data: we have a large number of predictive variables – from turnovers, steals, blocks, fouls, three-pointers, rebounds, and others, there are many variables we can use to determine the winner of each game. There are likely variables that are not very significant, and we can reduce the number of variables we have by using PCA or calculating the importance of the variables with a decision tree model and using basketball-intuition combined with the results to choose which combination of variables would likely yield the most accurate predictions.

We would like to predict the outcome of each game, and use a binary classifier to predict the winner and loser of each. Some models that can do this for us are logistic regression, decision tree classifiers, and neural networks using a multilayer perceptron.

Logistic regression could yield the likelihood of each team being named the winner of the match, and choosing the team with the highest probability. This would provide us an empirical way to calculate the probability of winners, and be able to be applied to every match in the bracket with ease.

A Tensorflow network is also an attractive option. A neural net could use the large amount of variables we have better than most other models. Creating a neural net model would let us predict outcomes of games very quickly and efficiently, and let us know the weights of each variable.

Whichever method we chose, we will obviously have to do a large amount of cross-validation and model tweaking to find the best model for predicting the winning team. Given that early upsets will yield large errors in a model that predicts the whole bracket, it may be beneficial to train our model and do cross-validation on the level of individual games, bracket stages, and the entire bracket. Training our accuracy by using parts of or the entire bracket would make incorrect classifications more costly, perhaps allowing us to tune hyperparameters for better predictions – though this may also cause us to overfit to the early games in the bracket. Which method yields higher total-bracket accuracy remains to be seen.

Finding the correct variables to use is also one of the larger challenges of this dataset. With so many to choose from, we will likely end up using domain knowledge combined with empirical results to choose the best combination. One blog describes coming up with a “win-loss” ratio by using “offensive-efficiency” and “defensive-efficiency” as the predictive variables3. Creating simple variables like these that can hold the information of many different variables we have is likely to be beneficial, as we suspect we will have a problem with collinearity if we use all of the variables.

References:

1. https://www.kaggle.com/c/march-machine-learning-mania-2017/data
2. <http://onlinelibrary.wiley.com/wol1/doi/10.1002/nav.20170/abstract>
3. <https://www.ibm.com/blogs/business-analytics/mens-basketball-bracket/>