**CS 235 Final**

**Authors: Hovanes Keseyan, William Siu**

This CS 235 Final is a collaborative effort and we decided on to take Option 3. We decided to use NBA statistics compiled over the years found on the internet. Using an online database source (dougstats.com basketball statistics) and massaging the data to fit our needs, we were able to compile the following results.

Originally we wanted to use the data to be able to separate into five unique branches/classes, but there were far too many branches. The data was too unrefined for our specific needs. We kept track on specific changes we made to the data to ensure that all changes and findings were recorded. The results will be separated into multiple steps listed below:

**Step 1:**

**Changes made:**

* Used only 2016 season
* Removed players with less than 500mins played
* Removed Player name and Team columns
* Change Positions: PG =1, SG=2, SF=3, PF=4, C=5

We wanted to do a comparison using the various methodology we learned in class. We approached it at different levels to ensure we were able to get similar results. Using the Nearest Neighbor method, and using training and test set (350 total players, and using 99 players as the training set) we were able to obtain the following results:

The dataset you tested has 5 classes

The training set is of size 99, and the test set is of size 350.

The time series are of length 17

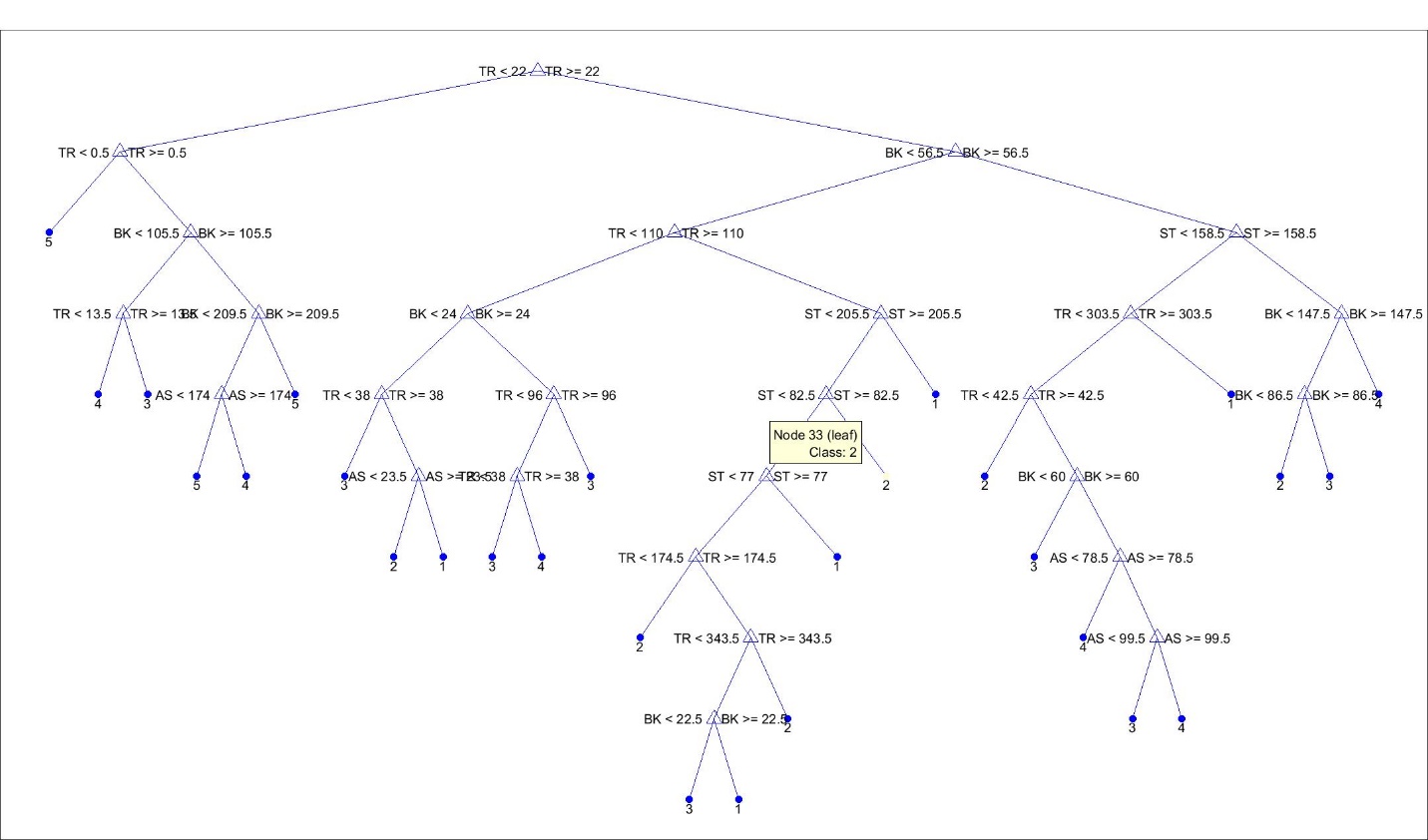
The error rate was 0.46571

Next we approached it using the Decision Tree, to be able to distinguish the different classes and see how many different branches the results would reveal.



*Figure 1: Histogram of number of players per position*

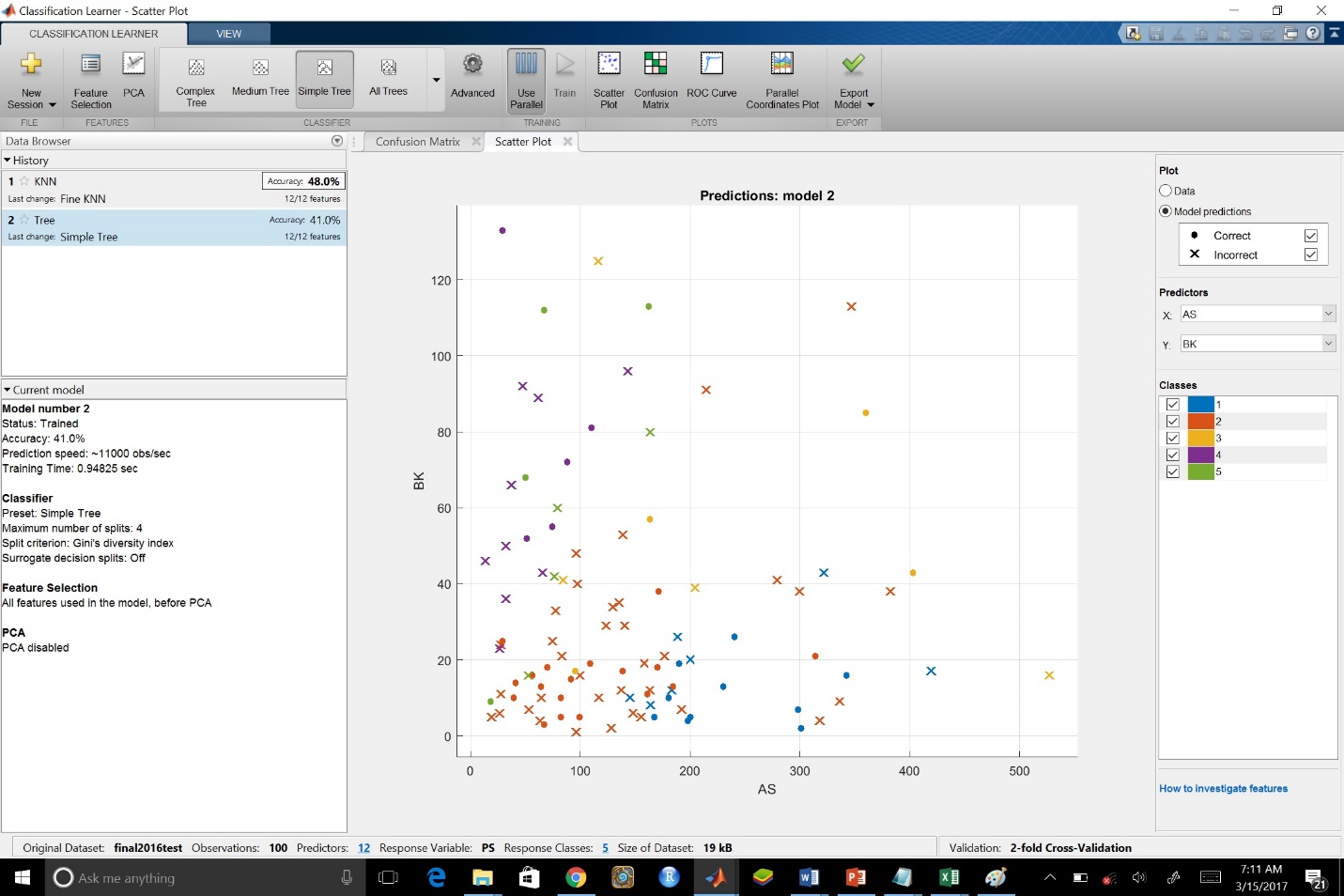
The histogram makes some sense because NBA teams tend to have less centers (position 5) on their team than the rest of the position.



*Figure 2: Decision Tree after Step 1*

Figure 1 identities the number of players at different positions using the histogram. Figure 2 identifies five noticeable branches, and upon looking more closely, we were able to distinguish them as different player positions. While the data is relatively raw, we can see that the decision tree started with TR (total rebounds) and worked downwards. However, the results don’t accurately portray how we expect typical NBA positions to break down. We have classes listed all over the decision tree, so the data doesn’t seem accurate.

We also tried the built-in tool that MATLAB provided called the Classification Learner. We are using this tool as a check and balance, to ensure our results are realistic and accurate. Using the built in classification tree and KNN tool, we obtained the following:



*Figure 3: MATLAB built-in Classification Leaner tool with scatter plot*

This tool allows us to decipher the data and get a visual of how our trends would look like. By selecting different predictors (Figure 3 selects AS and BK for the X and Y respectively), we were able to display the above scatter plot. We can also select the different classes (refer to top Step 1 Changes Made) and have it shown that specific classes do cluster together with a few outliers.

Also by using this tool we can see the KNN has an accuracy of 48%, similar to the error rate we obtained using our own nearest neighbor method we had used earlier (error rate was approximately 0.47).

Overall, we noticed there could be improvements to be made to the data. Our second approach to the data is documented below.

**Step 2:**

**Changes Made:**

* Refined data by removing starts and season columns
* Divided the stats by Games played

During Step 1 we noticed that there were still many unnecessary columns so we removed the Starts and Season columns from the data. We also decided to divide the stats by Games Played to display a more accurate analysis of players’ statistics. That way players’ data are not inflated as a result of the volume of their play time, since each position theoretically should receive equal play time.

Using the Nearest Neighbor approach again, had the following result:

The dataset you tested has 5 classes

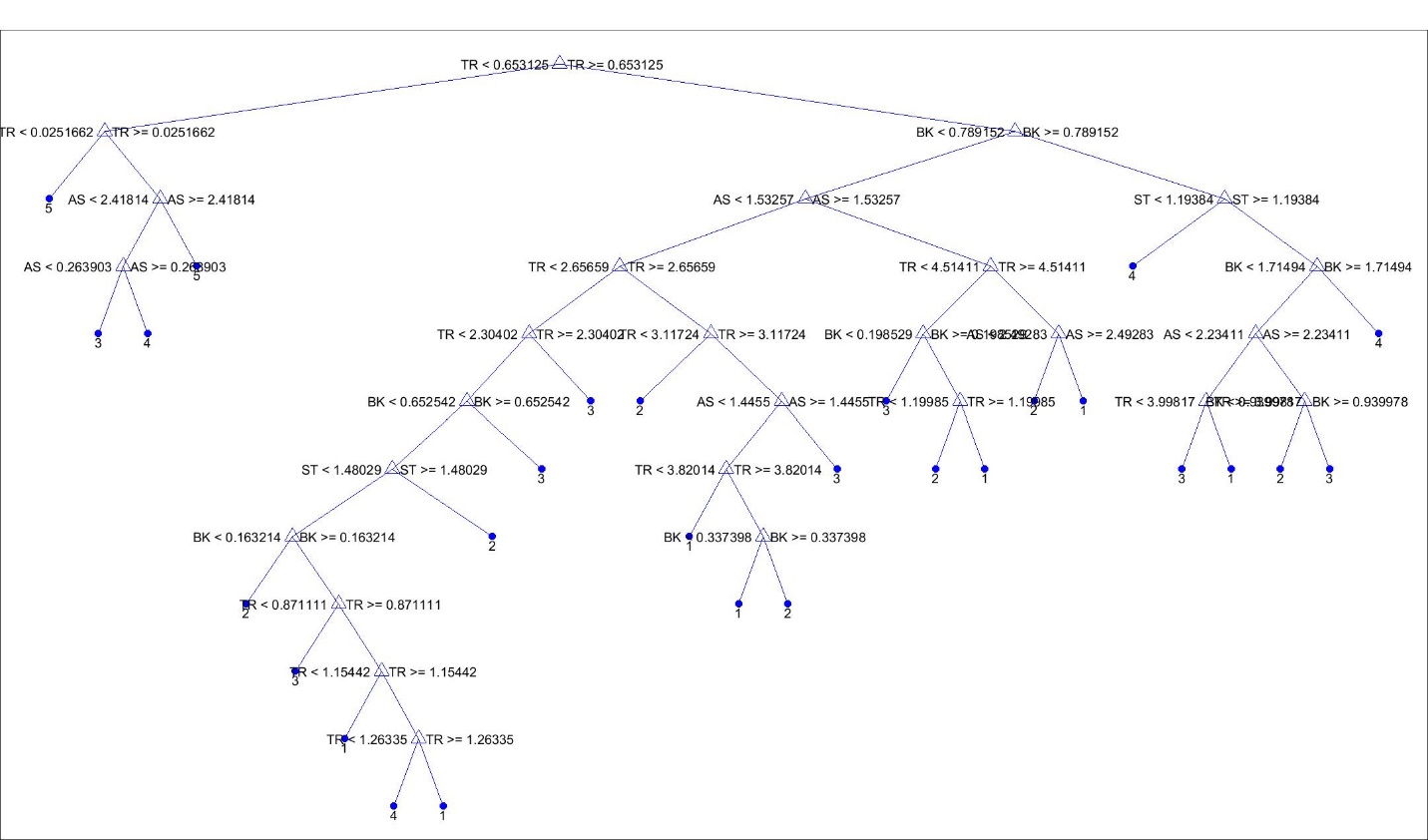
The training set is of size 99, and the test set is of size 350.

The time series are of length 15

The error rate was 0.56

This seems to reflect that the changes made have resulted in more errors, therefore not really improving the overall effects.

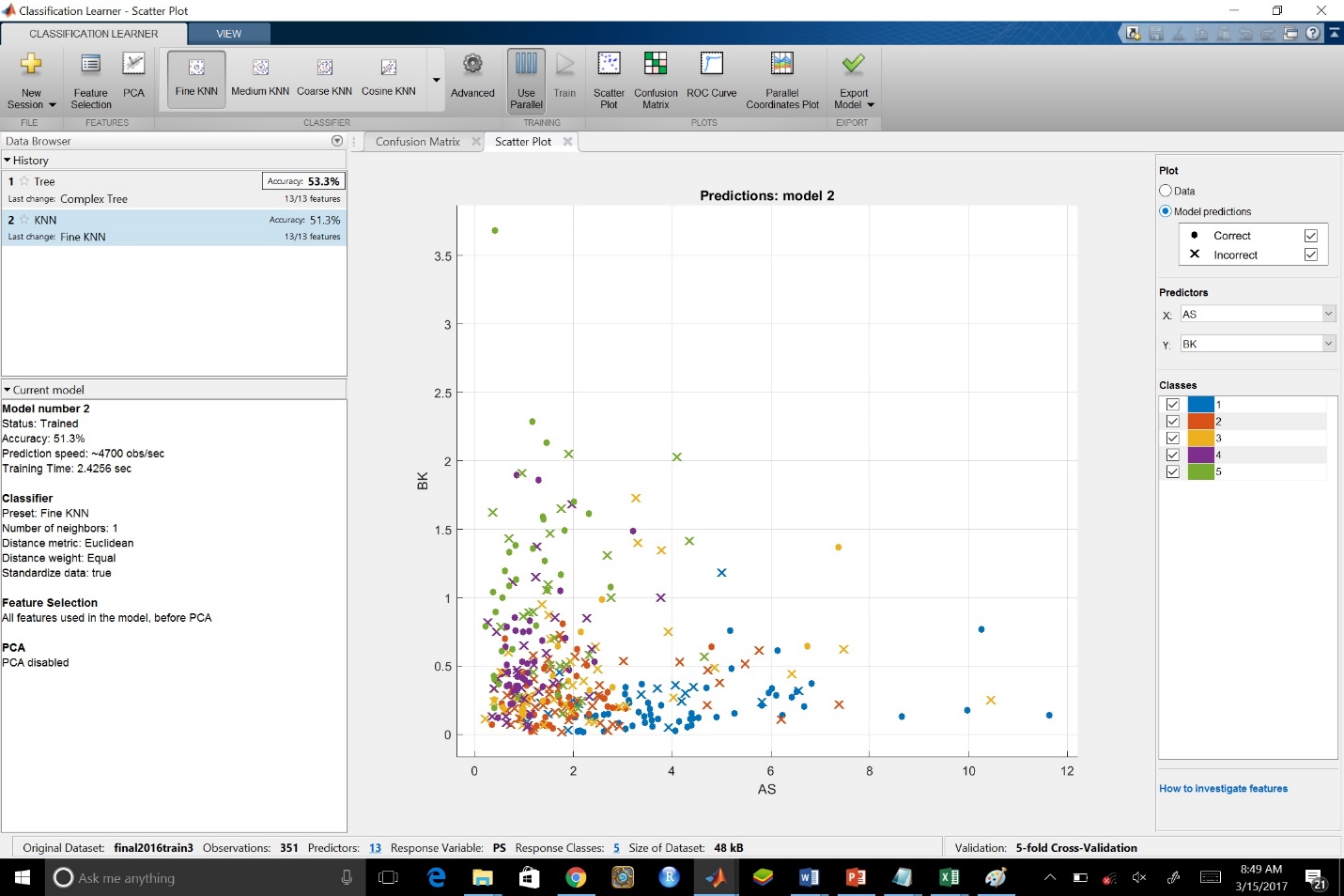
Using the Decision Tree method, we obtained the following results:



*Figure 4: Decision Tree after Step 2*

Figure 4 listed above shows there’s a different trend towards how the decision tree was created. It first identifies the Total Rebound columns and then trends down to Blocks, Assists and Steals. Using the decision tree method with the refinements, it’s still not an accurate portrayal that we would like to see. It seems there’s not a general clustering/branching to reflect just the player positions easily using this method.

Using the Classification Learner tool with the Step 2 changes, we obtained the following results:



*Figure 5: Classification Learner after Step 2*

Using the Classification Learner tool, we were able to come up with a visual using scatter plots. By playing with the Class fields again, we can see that the different classes accurately cluster to specific regions based on BLK and AST categories. Using this approach, we can see there are outliers with statistics far higher than their typical positions would normally be. The AST and BLK X/Y portrayal was used because they best reflect how players’ statistics based on their positions. Realistically taller frontcourt players would be the 4/5 (Power Forward and Centers) and would typically have the most blocks. Backcourt players 1/2 (Point Guard and Shooting Guard) would have more assists. There are outliers for each positions with more all-around talents which would prove this theory wrong, but overall the above scatter plot gives a good general visual of Classes/player positions.

By using the KNN nearest neighbor method built-in we show an accuracy of 51.3%, again relatively close to our error rate of 0.56. Ideally these numbers should be closer, but this method could be different from what we had used in class.

**Step 2:**

**Changes Made:**

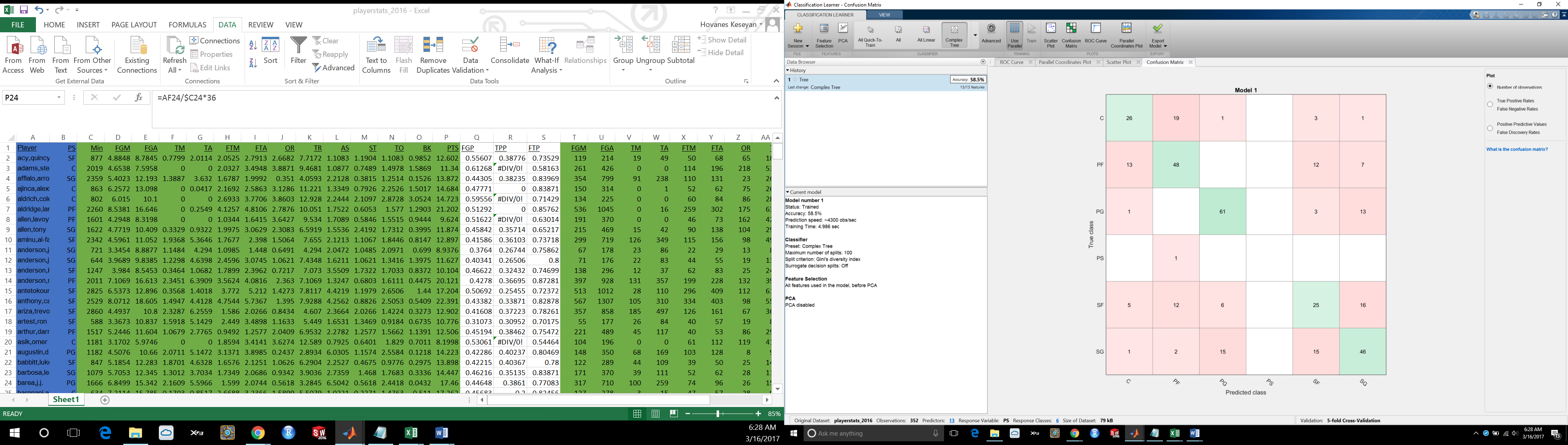
* Divided statistics based on minutes played instead of games played
* Tried new predictor fields by combining some previously used fields
* Spent more time with the Classification Learner to explore best model

Dividing total statistics by games played was an improvement over using season totals, and is generally ideal for identifying the best players because it accounts for efficiency over a large volume, but it is not the most ideal for determining a player’s position. As mentioned earlier, each position should theoretically see the exact same amount of playing time. As a result, we will divide the season totals based on minutes played, not games played. In order to work with more familiar numbers, we will take statistics per 36 minutes played, instead of per minute (because the numbers would be too tiny and unfamiliar, and per-36 is a fairly used industry standard in the sport).

Using our knowledge of basketball players and their positions, we decided to brainstorm some possible combinations of statistical fields that could help distinguish player positions. This involved dividing shot makes by shot attempts for field goals, free throws, and three-pointers respectively to obtain percentages. We also created the ratios of some of the statistics, in order to target some issues with distinguishing between positions. Table 1 below describes the issue, the statistic that could solve it, and the reasoning behind it.

*Table 1: Reasoning behind creation of new predictors*

|  |  |  |
| --- | --- | --- |
| Issue | New statistic | Reasoning |
| Guards with high rebounds | Offensive Rebounds per Total Rebound | Guards with high rebounds get more of their rebounds defensively |
| PF/C with high three-pointers | Three-pointers Made per Assist | PF/C with high three-pointers will fall behind on assists |
| Steals should distinguish PGs, filter out defense-oriented players | Field Goals Attempted per Steal | Defense-oriented players with high steals attempt less field goals |



*Figure 6: Confusion Matrix*

The Confusion Matrix is a great tool that summarizes how often each class is being confused with each other class. This helped us determine which cases we need to target, and helped us obtain the information in Table 1 above.

Based on previous steps, we can trust the Classification Learner to give us a similar acceptance rate to the code we had been using. With this information, we opened up the Classification Learner to test across a broad range of models and return he best one. This also involved using the Principal Components Analysis feature and the Parallel Coordinates Plot to help guide us towards the most accurate model.

The best model we could export was a Weighted KNN model with 61.6% accuracy. This gives us an error rate of 38.4%, which is significantly better than the values we were getting earlier with nearest neighbor, making our efforts to combine some fields a success. The predictor fields used in the final model were:

* Assists
* Blocks
* Total Rebounds
* Three-pointers Attempted
* Field Goal Percentage\*
* Offensive Rebounds per Total Rebound\*
* Three-pointers Made per Assist\*
* Field Goals Attempted per Steal\*

\*(combined field)

**Summary:**

Overall, from our observations, the usage of raw data of a single season of NBA player data is very hard to analyze. There are so many refinements that need to be made in order to gain an accurate portrayal and predictions of players. Using the Nearest Neighbor, Decision Tree, and built-in Classification Learner tool, we were able to interpret the data under different features. Each method revealed a different outlook and revealed additional improvements that needed to be made. The clearest methodology was the Classification Learner, which provided a visual scatter plot that accurately clustered by player positions with outliers. The decision tree ideally would separate out the player positions into unique branches based on the predictors. However, the decision tree results didn’t accurately display by player positions, but did show unique branching based on the data provided. The branching didn’t branch out as intended, but it did give us a portrayal of how players may differ despite their intended positions. Additional refinements of the data could be done in order to improve these decision tree. Finally the nearest neighbor method provide the error rates, which had gone up, despite the refinements. This could represent that the initial first step run of the NN method was more of a wild estimate, and doesn’t really provide us with any unique information. Overall the NN method along with the KNN (with 5 folds) didn’t really show that much difference, but provided us with a rough estimate that could be improved again with refinements to the data.

We were able to apply a lot of data mining techniques that we learned from class, such as preventing overfitting by having too many similar predictor fields, and combining some fields to obtain more relevant predictors (one of the combined fields made as much as a 4% different in accuracy rate). While we were not able to make as accurate a classification as we had hoped, we now understand the steps that a data scientist takes to improve the accuracy of their classifier.

This project allowed us to see how statisticians can play a vital role in sports. How each sports team and team physicians use these data will be entirely different, but we can see statisticians in a different light. It gives us a greater appreciation into the work that they do, and how much data is affecting not just the sports world, but our every-day lives.