Quarter 1 Project

Predicting NBA All-Stars

Gus Simanson and Lalit Boyapati
Thomas Jefferson High School for Science and Technology
Dr. Yilmaz
September 23, 2024

Table of Contents

Part 1 - Statement of Project Goal

Every February, the NBA selects 24 of their star players to play in an all-star game: Eastern Conference vs. Western Conference. These players are chosen through a vote: 50% is decided by fan vote, 25% by a current player vote, and 25% by a select group of NBA representatives. This game showcases the talent that the league has to offer and is always a spectacle to watch.

This leads us to an interesting question: can we predict which players will be named all-stars prior to the vote? Our goal is to retrieve data from the 2010-2023 seasons of all NBA players containing their individual statistics and train a model to predict whether or not a player will be designated as an all-star. We will also determine which metrics are most influential for this designation. Through this research, we will allow fans to make a more educated vote when the time comes and for players to have a better understanding on what stats they need to maximize to be in the running for one of those coveted spots.

Part 2 - Description of Dataset

We gathered our data from stats.nba and basketball-reference. Stats.nba provided us with the historical statistics of all NBA players, and basketball-reference.com gave us the list of named all-Stars. After scraping stats.nba for "traditional" statistics, we were able to construct a blanket dataset containing the players' statistics, and then we manually put in whether or not they were named an all-star.

Code to Web Scrape:

```
import requests
import pandas as pd
import time
from google.colab import files

url = 'https://stats.nba.com/stats/leaguedashplayerstats'

header = {
   'Host': 'stats.nba.com',
   'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36',
   'Accept': 'application/json, text/plain, */*',
   'Accept-Language': 'en-US,en;q=0.9',
   'Referer': 'https://www.nba.com/',
   'Origin': 'https://www.nba.com',
   'Connection': 'keep-alive'
}

all_data = []

def scrape_nba_stats_for_season(season):
   params = {
```

```python
        'College': '',
        'Conference': '',
        'Country': '',
        'DateFrom': '',
        'DateTo': '',
        'Division': '',
        'DraftPick': '',
        'DraftYear': '',
        'GameScope': '',
        'GameSegment': '',
        'Height': '',
        'LastNGames': '0',
        'LeagueID': '00',
        'Location': '',
        'MeasureType': 'Base',
        'Month': '0',
        'OpponentTeamID': '0',
        'Outcome': '',
        'PORound': '0',
        'PaceAdjust': 'N',
        'PerMode': 'PerGame',
        'Period': '0',
        'PlayerExperience': '',
        'PlayerPosition': '',
        'PlusMinus': 'N',
        'Rank': 'N',
        'Season': season,
        'SeasonSegment': '',
        'SeasonType': 'Regular Season',
        'ShotClockRange': '',
        'StarterBench': '',
        'TeamID': '0',
        'TwoWay': '0',
        'VsConference': '',
        'VsDivision': '',
        'Weight': ''
    }

    response = requests.get(url, headers=header, params=params)

    if response.status_code == 200:
        data = response.json()

        headers = data['resultSets'][0]['headers']
        rows = data['resultSets'][0]['rowSet']

        df = pd.DataFrame(rows, columns=headers)
        df['Season'] = season

        all_data.append(df)

        print(f"Data for season {season} successfully retrieved")

    else:
```

```
            print(f"Failed to retrieve data for season {season}. Status
code: {response.status_code}")

      seasons = [f'{year}-{str(year+1)[2:]}' for year in range(2010,2024)]

      for season in seasons:
         scrape_nba_stats_for_season(season)

      combined_df = pd.concat(all_data)

      combined_csv_name = 'nba_player_stats_all_seasons.csv'
      combined_df.to_csv(combined_csv_name, index=False)

      files.download(combined_csv_name)
```

Our dataset included 68 attributes, but at first glance the 27 most notable seem to be:

Age: Age of player during that season
GP: Games Played
W: Wins
L: Losses
MIN: Average minutes played a game
PTS: Average points a game
FGM: Field Goals Made per game – the number of field goals made by the player on average per game. This is any shot or tap in besides a free throw.
FGA: Field Goals Attempted - the number of field goals attempted by the player on average per game.
FG_PCT: Field Goal Percentage - the percentage of field goals made per game by the player on average.
FG3M: Three Pointers Made - the number of three pointers made by the player on average per game.
FG3_PCT: Three Pointer Percentage - the percentage of three pointers made by the player on average per game
FT_PCT: Free Throw Percentage - the percentage of free throws made by the player on average per game.
OREB: Offensive Rebounds per game - the number of offensive rebounds grabbed by the player on average per game.
DREB: Defensive Rebounds per game - the number of defensive rebounds grabbed by the player on average per game.
AST: Assists - the number of assists passed by the player on average per game.
TOV: Turnovers - the number of turnovers caused by the player on average per game.
STL: Steals - the number of steals forced by the player on average per game.
BLK: Blocks - the number of shots blocked by the player on average per game.
PF: Personal Fouls - the number of personal fouls committed by the player on average per game.
NBA_FANTASY_POINTS: The number of fantasy points generated by the player on average per game. Fantasy points are generated using a variety of stats.

DD2: Double-Doubles - the number of games in which a player achieves double digits in two statistical categories
TD3: Triple-Doubles - the number of games in which a player achieves double digits in three statistical categories
PLUS_MINUS: The point differential when a player is on the court.

And our class is:
All_Star_Selection: Whether or not player was named All-Star

These are not necessarily the attributes we will end up using, most likely the list will become much shorter. We will preprocess and undergo attribute selection to craft the strongest possible dataset to build the model from.

The dimension of our dataset is 68 with a total of 7190 instances. Since we scraped our data directly from stats.nba.com there are little to no missing values in the dataset. Our data is very heavily skewed as there are only 24 all-star players per year, while everyone else is a non-all-star. Therefore, the total 7190 instances only contain 538 all-star selections.

Part 3 - Pre-Processing

3.1 Missing Attributes:
As mentioned previously, we will not have to fill in missing values as the data was gathered from a comprehensive source.

3.2 Unrelated Attributes
We reviewed the attributes in our dataset and identified those that were irrelevant to the model. Attributes like Player ID, Player Name, Nickname, Team ID, WNBA Fantasy Rank, Season (year), and Team Abbreviation were determined to be unnecessary for predicting whether a player becomes an All-Star, even without performing attribute selection.

3.3 Derived Attributes
We discovered several derived attributes—such as games played, which can be calculated from wins and losses, REB (rebounds), which can be derived from adding DREB and OREB, Win PCT, which can be derived from wins divided by the total number of games, FT PCT, which can be derived from made and attempted free throws, FG PCT, which can be derived from made and attempted non-free throws, and FG3 PCT, which can be derived from made and attempted three point shots. We removed all of these from the dataset.

3.4 Normalization
Many of the attributes contain vastly different ranges of values, as the rank attributes can be in the 100's while blocks and steals typically stayed smaller. There were not many outliers within the data, so we decided to do min-max normalization on all of the remaining attributes because all of them had unique and wide ranges.

3.5 Numeric to Nominal
We had to switch the class attribute, All Star Selection, from numeric to nominal in order to apply attribute selection and SMOTE later on in this process.

3.6 SMOTE
There is a very large class imbalance in the dataset , with only 538 of the 7190 instances being all-stars (7%). To rectify this, we applied the Synthetic Minority Oversampling Technique (SMOTE, which resamples a dataset to add more instances of the minority class. SMOTE works through picking a random instance from the minority class, identifying a user specified number of nearest neighbors (typically five), and then for one randomly selected neighbor, it takes the difference between the instance and that neighbor, multiplies the difference by a random number between zero and one, and adds that difference to the instance. This provides you with a synthetic instance in your minority class. We ran SMOTE twice on the data, and achieved a more balanced ratio of 2152 all-stars to 8804 instances (24%).

3.7 Remaining Attributes

1. AGE
2. W
3. L
4. MIN
5. FGM
6. FGA
7. FG3M
8. FG3A
9. FTM
10. FTA
11. OREB
12. DREB
13. AST
14. TOV
15. STL
16. BLK
17. BLKA
18. PF
19. PFD
20. PTS
21. PLUS_MINUS
22. NBA_FANTASY_PTS
23. DD2
24. TD3
25. GP_RANK
26. W_RANK
27. L_RANK
28. W_PCT_RANK
29. MIN_RANK
30. FGM_RANK
31. FGA_RANK
32. FG_PCT_RANK
33. FG3M_RANK
34. FG3A_RANK
35. FG3_PCT_RANK
36. FTM_RANK
37. FTA_RANK
38. FT_PCT_RANK
39. OREB_RANK
40. DREB_RANK
41. REB_RANK
42. AST_RANK
43. TOV_RANK
44. STL_RANK
45. BLK_RANK
46. BLKA_RANK
47. PF_RANK
48. PFD_RANK
49. PTS_RANK
50. PLUS_MINUS_RANK
51. NBA_FANTASY_PTS_RANK
52. DD2_RANK
53. TD3_RANK
54. All_Star_Selection

Part 4 - Attribute Selection

After importing the preprocessed dataset into Weka, we decided to run the CorrelationAttributeEval, CfsSubsetEval, InfoGainAttributeEval, and OneRAttributeEval. Weka's output for each is displayed below. We also created our own dataset, with attributes we felt to be the most appropriate.

4.1 CorrelationAttributeEval

CoorelationAttributeEval is one of the attribute selection algorithms we chose to run on our dataset. This calculates the Pearson correlation coefficient between each feature and the class. This then ranks the features in a listed manner as shown below in the screenshot:

```
Attribute Evaluator (supervised, Class (nominal): 54 All_Star_Selection):
        Correlation Ranking Filter
Ranked attributes:
 0.062185    1 AGE
 0.05316    21 PLUS_MINUS
 0.050227   50 PLUS_MINUS_RANK
 0.031603   26 W_RANK
 0.031529   45 BLK_RANK
 0.025575   37 FTA_RANK
 0.024148    2 W
 0.023675   28 W_PCT_RANK
 0.023367   38 FT_PCT_RANK
 0.022805   39 OREB_RANK
 0.021037   44 STL_RANK
 0.0207     40 DREB_RANK
 0.019931   41 REB_RANK
 0.019801   15 STL
 0.019262   35 FG3_PCT_RANK
 0.018113   36 FTM_RANK
 0.017732    5 FGM
 0.015941   20 PTS
 0.014166   53 TD3_RANK
 0.013723   13 AST
 0.013652    6 FGA
 0.012685   47 PF_RANK
 0.012499   46 BLKA_RANK
 0.012222   14 TOV
 0.011615   10 FTA
 0.011351   25 GP_RANK
 0.010839   49 PTS_RANK
 0.010758   30 FGM_RANK
 0.010091   51 NBA_FANTASY_PTS_RANK
 0.009865   34 FG3A_RANK
 0.009652   48 PFD_RANK
 0.009615   17 BLKA
 0.009125   24 TD3
 0.008928   22 NBA_FANTASY_PTS
 0.00881    27 L_RANK
 0.008561    9 FTM
 0.008532   31 FGA_RANK
 0.008373   33 FG3M_RANK
 0.008026   32 FG_PCT_RANK
 0.007851   52 DD2_RANK
 0.007505   23 DD2
 0.005311   43 TOV_RANK
 0.005166    7 FG3M
 0.005092   18 PF
 0.00505    16 BLK
 0.004934    3 L
 0.004608   42 AST_RANK
 0.003905   11 OREB
 0.003639   19 PFD
 0.003425   12 DREB
 0.002834    4 MIN
 0.001585    8 FG3A
 0.000618   29 MIN_RANK

Selected attributes: 1,21,50,26,45,37,2,28,38,39,44,40,41,15,35,36,5,20,53,13,6,47,46,14,10,25,49,30,51,34,48,17,24,22,27,9,31,33,32,52,23,43,7,18,16,3,42,11,19,12,4,8,29 : 53
```

By setting out correlation value cutoff at 0.02 we can see the relevant attributes based on CorrelationAttributeEval are [AGE, PLUS_MINUS, PLUS_MINUS_RANK, W_RANK, BLK_RANK, FTA_RANK, W, W_PCT_RANK, FT_PCT_RANK, OREB_RANK, STL_RANK, DREB_RNK]

4.2 CfsSubsetEval

CfsSubsetEval is one of the attribute selection algorithms we chose to run on our dataset. CfsSubsetEval works by evaluating the degree of redundancy among features associated with the class. It selects features that are highly correlated with the target variable but not correlated with each other.

```
              NBA_FANTASY_PTS
              DD2
              TD3
              GP_RANK
              W_RANK
              L_RANK
              W_PCT_RANK
              MIN_RANK
              FGM_RANK
              FGA_RANK
              FG_PCT_RANK
              FG3M_RANK
              FG3A_RANK
              FG3_PCT_RANK
              FTM_RANK
              FTA_RANK
              FT_PCT_RANK
              OREB_RANK
              DREB_RANK
              REB_RANK
              AST_RANK
              TOV_RANK
              STL_RANK
              BLK_RANK
              BLKA_RANK
              PF_RANK
              PFD_RANK
              PTS_RANK
              PLUS_MINUS_RANK
              NBA_FANTASY_PTS_RANK
              DD2_RANK
              TD3_RANK
              All_Star_Selection
Evaluation mode:    evaluate on all training data



=== Attribute Selection on all input data ===

Search Method:
        Greedy Stepwise (forwards).
        Start set: no attributes
        Merit of best subset found:    0.294

Attribute Subset Evaluator (supervised, Class (nominal): 54 All_Star_Selection):
        CFS Subset Evaluator
        Including locally predictive attributes

Selected attributes: 1,7,11,14,15,16,17,21,53 : 9
                     AGE
                     FG3M
                     OREB
                     TOV
                     STL
                     BLK
                     BLKA
                     PLUS_MINUS
                     TD3_RANK
```

This created a dataset with [AGE, FG3M, OREB, TOV, STL, BLK, BLKA, PLUS_MINUS, TD3_RANK] as the attributes.

4.3 InfoGainAttributeEval

InfoGainAttributeEval determines how well a given attribute separates the training examples according to their class labels. The higher the ranking means the attribute is more informative for classification.

```
        Information Gain Ranking Filter

Ranked attributes:
 0.42707    1 AGE
 0.3909    53 TD3_RANK
 0.38878   11 OREB
 0.37236   14 TOV
 0.3652    15 STL
 0.36494   17 BLKA
 0.35663    7 FG3M
 0.32901   16 BLK
 0.28443   19 PFD
 0.255      9 FTM
 0.24841   10 FTA
 0.23932   52 DD2_RANK
 0.14831   23 DD2
 0.10132    8 FG3A
 0.02734    2 W
 0.02479   24 TD3
 0.0242     6 FGA
 0.00982   21 PLUS_MINUS
 0.00907    3 L
 0.00875   25 GP_RANK
 0.00813   28 W_PCT_RANK
 0.00805   35 FG3_PCT_RANK
 0.00793   50 PLUS_MINUS_RANK
 0.0071    38 FT_PCT_RANK
 0.00661   27 L_RANK
 0.00511   34 FG3A_RANK
 0.0043    45 BLK_RANK
 0.00422   44 STL_RANK
 0.0034    29 MIN_RANK
 0.00318   47 PF_RANK
 0.00312    5 FGM
 0.00298   26 W_RANK
 0.00289   33 FG3M_RANK
 0.00275   31 FGA_RANK
 0.00266   37 FTA_RANK
 0.00212   36 FTM_RANK
 0.00209   39 OREB_RANK
 0.00196   46 BLKA_RANK
 0.0019    42 AST_RANK
 0.0018    40 DREB_RANK
 0         49 PTS_RANK
 0          4 MIN
 0         41 REB_RANK
 0         43 TOV_RANK
 0         48 PFD_RANK
 0         22 NBA_FANTASY_PTS
 0         20 PTS
 0         18 PF
 0         30 FGM_RANK
 0         32 FG_PCT_RANK
 0         12 DREB
 0         13 AST
 0         51 NBA_FANTASY_PTS_RANK

Selected attributes: 1,53,11,14,15,17,7,16,19,9,10,52,23,8,2,24,6,21,3,25,28,35,50,38,27,34,45,44,29,47,5,26,33,31,37,36,39,46,42,40,49,4,41,43,48,22,20,18,30,32,12,13,51 : 53
```

By setting the cutoff value to 0.2, we were left with a dataset containing the attribute set of [AGE, TD3_RANK, OREB, TOV, STL, BLKA, FG3M, BLK, PFD, FTM, FTA, DD2_RANK]

4.4 OneRAttributeEval

OneRAttributeEval in WEKA evaluates attributes by creating one-rule classifiers based on each attribute and measuring their classification error rates. It selects the attribute that produces the lowest error rate.

```
        Minimum bucket size for OneR: 6

Ranked attributes:
92.48069    3 L
92.34439    2 W
92.26488    1 AGE
91.87869   18 PF
91.28805   12 DREB
91.12903   19 PFD
91.10632   11 OREB
90.90186   21 PLUS_MINUS
90.90186    8 FG3A
90.8905    14 TOV
90.811     53 TD3_RANK
90.62926   10 FTA
90.62926    5 FGM
90.59518   15 STL
90.51567    7 FG3M
90.36801   13 AST
90.27715    9 FTM
89.91368   17 BLKA
89.56156   16 BLK
88.43707   52 DD2_RANK
87.92594    6 FGA
86.21081   20 PTS
83.83689   23 DD2
81.07678   26 W_RANK
81.00863   27 L_RANK
80.57701    4 MIN
79.72512   22 NBA_FANTASY_PTS
79.57746   25 GP_RANK
77.38528   49 PTS_RANK
77.36256   35 FG3_PCT_RANK
77.28305   37 FTA_RANK
77.24898   41 REB_RANK
77.16947   34 FG3A_RANK
77.05588   43 TOV_RANK
76.99909   42 AST_RANK
76.99909   39 OREB_RANK
76.93094   48 PFD_RANK
76.93094   44 STL_RANK
76.93094   31 FGA_RANK
76.91958   45 BLK_RANK
76.89687   51 NBA_FANTASY_PTS_RANK
76.88551   36 FTM_RANK
76.88551   33 FG3M_RANK
76.87415   32 FG_PCT_RANK
76.85143   50 PLUS_MINUS_RANK
76.82871   38 FT_PCT_RANK
76.806     46 BLKA_RANK
76.71513   30 FGM_RANK
76.70377   29 MIN_RANK
76.70377   40 DREB_RANK
76.56747   24 TD3
76.56747   47 PF_RANK
76.28351   28 W_PCT_RANK

Selected attributes: 3,2,1,18,12,19,11,21,8,14,53,10,5,15,7,13,9,17,16,52,6,20,23,26,27,4,22,25,49,35,37,41,34,43,42,39,48,44,31,45,51,36,33,32,50,38,46,30,29,40,24,47,28 : 53
```

We decided to use a cutoff value of 90.8, leaving us with an attribute set of [L, W, AGE, PF, DREB, PFD, OREB, PLUS_MINUS, FG3A, TOV, TD3_RANK].

4.5 Set chosen by us

Based on our background knowledge about the NBA and player production, we chose a dataset with **[**W, FGM, PLUS_MINUS, PLUS_MINUS_RANK, BLK, STL, AST, FG3M, and MIN] as the attributes because in our experience as avid basketball fans we believe stellar performance in these attributes to be key in order to produce an elite basketball player.

Part 5 - Train, Test, Validation Split

After attribute selection, we split the data into a train, test, validation split using the code below. This split was 80% train, 10% test, and 10% validation.

Code to split:

```
from google.colab import files
import pandas as pd
from sklearn.model_selection import train_test_split

uploaded = files.upload()
file_name = next(iter(uploaded))
data = pd.read_csv(file_name)
```

```
train_data, temp_data = train_test_split(data, test_size=0.20,
random_state=42)

val_data, test_data = train_test_split(temp_data, test_size=0.50,
random_state=42)

print(f"Training set: {train_data.shape}")
print(f"Validation set: {val_data.shape}")
print(f"Test set: {test_data.shape}")

train_data.to_csv('train_data.csv', index=False)
val_data.to_csv('val_data.csv', index=False)
test_data.to_csv('test_data.csv', index=False)

files.download('train_data.csv')
files.download('val_data.csv')
files.download('test_data.csv')
```

Part 6 - Classification Models

For each of the datasets created through our attribute selection we created four classification models: NaiveBayes, J48, OneR, and Logistic.

NaiveBayes is a probabilistic classifier that assumes the presence of each feature is independent of the other features.

J48 is a decision tree algorithm that splits the data based on attribute values. It creates tree-structures that can handle both categorical and continuous data.

OneR is a simple, rule-based classifier that generates one rule for each predictor and selects the one that performs the best.

Logistic is a statistical model used for binary classification. It assumes a linear relationship between the independent variable and the log odds of the dependent variable.

6.1 CorrelationAttributeEval with NaiveBayes Classification

```
Time taken to build model: 0.03 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances         665               75.4824 %
Incorrectly Classified Instances       216               24.5176 %
Kappa statistic                         -0.0019
Mean absolute error                      0.3665
Root mean squared error                  0.4359
Relative absolute error                100.3156 %
Root relative squared error            102.9341 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.979    0.981    0.766      0.979    0.860      -0.004   0.550     0.817     0
                 0.019    0.021    0.222      0.019    0.036      -0.004   0.550     0.261     1
Weighted Avg.    0.755    0.756    0.639      0.755    0.667      -0.004   0.550     0.687

=== Confusion Matrix ===

   a   b   <-- classified as
 661  14 |   a = 0
 202   4 |   b = 1
```

6.2 CorrelationAttributeEval with J48 Classification

```
Number of Leaves  :     81

Size of the tree :      161


Time taken to build model: 0.41 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         803               91.1464 %
Incorrectly Classified Instances        78                8.8536 %
Kappa statistic                          0.724
Mean absolute error                      0.144
Root mean squared error                  0.2825
Relative absolute error                 39.3971 %
Root relative squared error             66.7216 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.988    0.340    0.905      0.988    0.945      0.742    0.842     0.910     0
                 0.660    0.012    0.944      0.660    0.777      0.742    0.842     0.789     1
Weighted Avg.    0.911    0.263    0.914      0.911    0.906      0.742    0.842     0.881

=== Confusion Matrix ===

   a   b   <-- classified as
 667   8 |   a = 0
  70 136 |   b = 1
```

6.3 CorrelationAttributeEval with OneR Classification

```
Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances         814               92.395  %
Incorrectly Classified Instances        67                7.605  %
Kappa statistic                          0.7607
Mean absolute error                      0.076
Root mean squared error                  0.2758
Relative absolute error                 20.8138 %
Root relative squared error             65.1217 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.325    0.910      1.000    0.953      0.783    0.837     0.910     0
                 0.675    0.000    1.000      0.675    0.806      0.783    0.837     0.751     1
Weighted Avg.    0.924    0.249    0.931      0.924    0.918      0.783    0.837     0.873

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
  67 139 |   b = 1
```

6.4 CorrelationAttributeEval with Logistic Classification

```
Time taken to build model: 0.19 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         675               76.6175 %
Incorrectly Classified Instances       206               23.3825 %
Kappa statistic                          0
Mean absolute error                      0.3628
Root mean squared error                  0.4237
Relative absolute error                 99.2866 %
Root relative squared error            100.0521 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.766      1.000    0.868      ?        0.522     0.799     0
                 0.000    0.000    ?          0.000    ?          ?        0.522     0.256     1
Weighted Avg.    0.766    0.766    ?          0.766    ?          ?        0.522     0.672

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
 206   0 |   b = 1
```

6.5 CfsSubsetEval with NaiveBayes Classification

```
Time taken to build model: 0.02 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances         675               76.6175 %
Incorrectly Classified Instances       206               23.3825 %
Kappa statistic                          0
Mean absolute error                      0.361
Root mean squared error                  0.4221
Relative absolute error                 98.7958 %
Root relative squared error             99.683  %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.766      1.000    0.868      ?        0.585     0.815     0
                 0.000    0.000    ?          0.000    ?          ?        0.585     0.302     1
Weighted Avg.    0.766    0.766    ?          0.766    ?          ?        0.585     0.695

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
 206   0 |   b = 1
```

6.6 CfsSubsetEval with J48 Classification

```
Number of Leaves  :     202

Size of the tree :      403


Time taken to build model: 0.14 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         794               90.1249 %
Incorrectly Classified Instances        87                9.8751 %
Kappa statistic                          0.7028
Mean absolute error                      0.1397
Root mean squared error                  0.3006
Relative absolute error                 38.2386 %
Root relative squared error             70.9906 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.967    0.316    0.909      0.967    0.938      0.711    0.831     0.907     0
                 0.684    0.033    0.865      0.684    0.764      0.711    0.831     0.765     1
Weighted Avg.    0.901    0.249    0.899      0.901    0.897      0.711    0.831     0.874

=== Confusion Matrix ===

   a   b   <-- classified as
 653  22 |   a = 0
  65 141 |   b = 1
```

6.7 CfsSubsetEval with OneR Classification

```
Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         815               92.5085 %
Incorrectly Classified Instances        66                7.4915 %
Kappa statistic                          0.7647
Mean absolute error                      0.0749
Root mean squared error                  0.2737
Relative absolute error                 20.5031 %
Root relative squared error             64.6339 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.320    0.911      1.000    0.953      0.787    0.840     0.911     0
                 0.680    0.000    1.000      0.680    0.809      0.787    0.840     0.755     1
Weighted Avg.    0.925    0.245    0.932      0.925    0.920      0.787    0.840     0.874

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
  66 140 |   b = 1
```

6.8 CfsSubsetEval with Logistic Classification

```
Time taken to build model: 0.13 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         675               76.6175 %
Incorrectly Classified Instances       206               23.3825 %
Kappa statistic                          0
Mean absolute error                      0.3625
Root mean squared error                  0.4225
Relative absolute error                 99.2191 %
Root relative squared error             99.7631 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.766      1.000    0.868      ?        0.549     0.820     0
                 0.000    0.000    ?          0.000    ?          ?        0.549     0.249     1
Weighted Avg.    0.766    0.766    ?          0.766    ?          ?        0.549     0.686

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
 206   0 |   b = 1
```

6.9 InfoGainAttributeEval with NaiveBayes Classification

```
Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances         675               76.6175 %
Incorrectly Classified Instances       206               23.3825 %
Kappa statistic                          0
Mean absolute error                      0.3664
Root mean squared error                  0.4266
Relative absolute error                100.2776 %
Root relative squared error            100.7404 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.766      1.000    0.868      ?        0.546     0.792     0
                 0.000    0.000    ?          0.000    ?          ?        0.546     0.275     1
Weighted Avg.    0.766    0.766    ?          0.766    ?          ?        0.546     0.671

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
 206   0 |   b = 1
```

6.10 InfoGainAttributeEval with J48 Classification

```
Number of Leaves  :     200

Size of the tree :      399


Time taken to build model: 0.16 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         792               89.8978 %
Incorrectly Classified Instances        89               10.1022 %
Kappa statistic                          0.6949
Mean absolute error                      0.1432
Root mean squared error                  0.3055
Relative absolute error                 39.1994 %
Root relative squared error             72.1507 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.967    0.325    0.907      0.967    0.936      0.703    0.826     0.906     0
                 0.675    0.033    0.863      0.675    0.757      0.703    0.826     0.741     1
Weighted Avg.    0.899    0.257    0.897      0.899    0.894      0.703    0.826     0.867

=== Confusion Matrix ===

   a   b   <-- classified as
 653  22 |   a = 0
  67 139 |   b = 1
```

6.11 InfoGainAttributeEval with OneR Classification

```
Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         815               92.5085 %
Incorrectly Classified Instances        66                7.4915 %
Kappa statistic                          0.7647
Mean absolute error                      0.0749
Root mean squared error                  0.2737
Relative absolute error                 20.5031 %
Root relative squared error             64.6339 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.320    0.911      1.000    0.953      0.787    0.840     0.911     0
                 0.680    0.000    1.000      0.680    0.809      0.787    0.840     0.755     1
Weighted Avg.    0.925    0.245    0.932      0.925    0.920      0.787    0.840     0.874

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
  66 140 |   b = 1
```

6.12 InfoGainAttributeEval with Logistic Classification

```
Time taken to build model: 0.15 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         675               76.6175 %
Incorrectly Classified Instances       206               23.3825 %
Kappa statistic                          0
Mean absolute error                      0.3646
Root mean squared error                  0.4249
Relative absolute error                 99.7754 %
Root relative squared error            100.3467 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.766      1.000    0.868      ?        0.501     0.784     0
                 0.000    0.000    ?          0.000    ?          ?        0.501     0.226     1
Weighted Avg.    0.766    0.766    ?          0.766    ?          ?        0.501     0.653

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
 206   0 |   b = 1
```

6.13 OneRAttributeEval with NaiveBayes Classification

```
Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances         675               76.6175 %
Incorrectly Classified Instances       206               23.3825 %
Kappa statistic                          0
Mean absolute error                      0.3612
Root mean squared error                  0.424
Relative absolute error                 98.8576 %
Root relative squared error            100.1282 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.766      1.000    0.868      ?        0.574     0.811     0
                 0.000    0.000    ?          0.000    ?          ?        0.574     0.307     1
Weighted Avg.    0.766    0.766    ?          0.766    ?          ?        0.574     0.693

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
 206   0 |   b = 1
```

6.14 OneRAttributeEval with J48 Classification

```
Number of Leaves  :     142

Size of the tree :      283


Time taken to build model: 0.13 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         795               90.2384 %
Incorrectly Classified Instances        86                9.7616 %
Kappa statistic                          0.7046
Mean absolute error                      0.1461
Root mean squared error                  0.2955
Relative absolute error                 39.9956 %
Root relative squared error             69.7857 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.970    0.320    0.908      0.970    0.938      0.714    0.866     0.929     0
                 0.680    0.030    0.875      0.680    0.765      0.714    0.866     0.768     1
Weighted Avg.    0.902    0.252    0.901      0.902    0.898      0.714    0.866     0.892

=== Confusion Matrix ===

   a   b   <-- classified as
 655  20 |   a = 0
  66 140 |   b = 1
```

6.15 OneRAttributeEval with OneR Classification

```
Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         814               92.395  %
Incorrectly Classified Instances        67                7.605  %
Kappa statistic                          0.7607
Mean absolute error                      0.076
Root mean squared error                  0.2758
Relative absolute error                 20.8138 %
Root relative squared error             65.1217 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.325    0.910      1.000    0.953      0.783    0.837     0.910     0
                 0.675    0.000    1.000      0.675    0.806      0.783    0.837     0.751     1
Weighted Avg.    0.924    0.249    0.931      0.924    0.918      0.783    0.837     0.873

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
  67 139 |   b = 1
```

6.16 OneRAttributeEval with Logistic Classification

```
Time taken to build model: 0.15 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         675               76.6175 %
Incorrectly Classified Instances       206               23.3825 %
Kappa statistic                          0
Mean absolute error                      0.3626
Root mean squared error                  0.4226
Relative absolute error                 99.2286 %
Root relative squared error             99.8049 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.766      1.000    0.868      ?        0.539     0.821     0
                 0.000    0.000    ?          0.000    ?          ?        0.539     0.241     1
Weighted Avg.    0.766    0.766    ?          0.766    ?          ?        0.539     0.685

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
 206   0 |   b = 1
```

6.17 Our Chosen Attributes with NaiveBayes Classification

```
Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0.01 seconds

=== Summary ===

Correctly Classified Instances         675               76.6175 %
Incorrectly Classified Instances       206               23.3825 %
Kappa statistic                          0
Mean absolute error                      0.363
Root mean squared error                  0.4224
Relative absolute error                 99.3353 %
Root relative squared error             99.743  %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.766      1.000    0.868      ?        0.562     0.819     0
                 0.000    0.000    ?          0.000    ?          ?        0.562     0.277     1
Weighted Avg.    0.766    0.766    ?          0.766    ?          ?        0.562     0.692

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
 206   0 |   b = 1
```

6.18 Our Chosen Attributes with J48 Classification

```
Number of Leaves  :     134

Size of the tree :      267


Time taken to build model: 0.16 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         783               88.8763 %
Incorrectly Classified Instances        98               11.1237 %
Kappa statistic                          0.6559
Mean absolute error                      0.1634
Root mean squared error                  0.3166
Relative absolute error                 44.7208 %
Root relative squared error             74.7704 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.970    0.379    0.894      0.970    0.930      0.670    0.796     0.893     0
                 0.621    0.030    0.865      0.621    0.723      0.670    0.796     0.698     1
Weighted Avg.    0.889    0.297    0.887      0.889    0.882      0.670    0.796     0.847

=== Confusion Matrix ===

   a   b   <-- classified as
 655  20 |   a = 0
  78 128 |   b = 1
```

6.19 Our Chosen Attributes with OneR Classification

```
Time taken to build model: 0.01 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         814               92.395  %
Incorrectly Classified Instances        67                7.605  %
Kappa statistic                          0.7607
Mean absolute error                      0.076
Root mean squared error                  0.2758
Relative absolute error                 20.8138 %
Root relative squared error             65.1217 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.325    0.910      1.000    0.953      0.783    0.837     0.910     0
                 0.675    0.000    1.000      0.675    0.806      0.783    0.837     0.751     1
Weighted Avg.    0.924    0.249    0.931      0.924    0.918      0.783    0.837     0.873

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
  67 139 |   b = 1
```

6.20 Our Chosen Attributes with Logistic Classification

```
Time taken to build model: 0.12 seconds

=== Evaluation on test set ===

Time taken to test model on supplied test set: 0 seconds

=== Summary ===

Correctly Classified Instances         675               76.6175 %
Incorrectly Classified Instances       206               23.3825 %
Kappa statistic                          0
Mean absolute error                      0.3649
Root mean squared error                  0.4236
Relative absolute error                 99.8554 %
Root relative squared error            100.0296 %
Total Number of Instances              881

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.766      1.000    0.868      ?        0.536     0.809     0
                 0.000    0.000    ?          0.000    ?          ?        0.536     0.242     1
Weighted Avg.    0.766    0.766    ?          0.766    ?          ?        0.536     0.677

=== Confusion Matrix ===

   a   b   <-- classified as
 675   0 |   a = 0
 206   0 |   b = 1
```

Part 7 - Performance Comparison/Results

When examining the results of the classifiers, we can generate the following table based upon their performance in Accuracy, True positive rate, and ROC Area.

Accuracy	NaiveBayes	J48	OneR	Logistic
CorrelationAttrib	75.4824	91.1464	92.395	76.6175
CfsSubsetEval	76.6175	90.1249	92.5085	76.6175
InfoGainAttribute	76.6175	89.8978	92.5085	76.6175
OneRAttributeEv	76.6175	90.2384	92.395	76.6175
Our Chosen At	76.6175	88.8763	92.395	76.6175

TP Rate	NaiveBayes	J48	OneR	Logistic
CorrelationAttrib	0.019	0.66	0.675	0
CfsSubsetEval	0	0.684	0.68	0
InfoGainAttribute	0	0.675	0.68	0
OneRAttributeEv	0	0.68	0.675	0
Our Chosen At	0	0.621	0.675	0

ROC Area	NaiveBayes	J48	OneR	Logistic
CorrelationAttrib	0.55	0.842	0.837	0.522
CfsSubsetEval	0.585	0.831	0.84	0.549
InfoGainAttribute	0.546	0.826	0.84	0.501
OneRAttributeEv	0.574	0.866	0.837	0.539
Our Chosen At	0.562	0.796	0.837	0.536

From this table, we generated a list of the 5 highest accuracies, true positive rates, and ROC areas:

Five Highest Accuracies:
1. CfsSubsetEval with OneR Classification - 92.5085%
1. InfoGainAttributeEval with OneR Classification - 92.5085%
2. CorrelationAttributeEval with OneR Classification - 92.395%
2. OneRAttributeEval with OneR Classification - 92.395%
2. Our Chosen Attributes with OneR Classification - 92.395%

Five Highest True Positive Rates:
1. CfsSubsetEval with J48 Classification - 0.684
2. CfsSubsetEval with OneR Classification - 0.68
2. InfoGainAttributeEval with OneR Classification - 0.68
2. OneRAttributeEval with J48 Classification - 0.68
5. InfoGainAttributeEval with J48 Classification - 0.675

Five Highest ROC Areas:
1. OneRAttributeEval with J48 Classification - 0.866
2. CorrelationAttributeEval with J48 Classification - 0.842
3. CfsSubsetEval with OneR Classification - 0.84
3. InfoGainAttributeEval with OneR Classification - 0.84

5. Our Chosen Attributes with One R Classification - 0.837

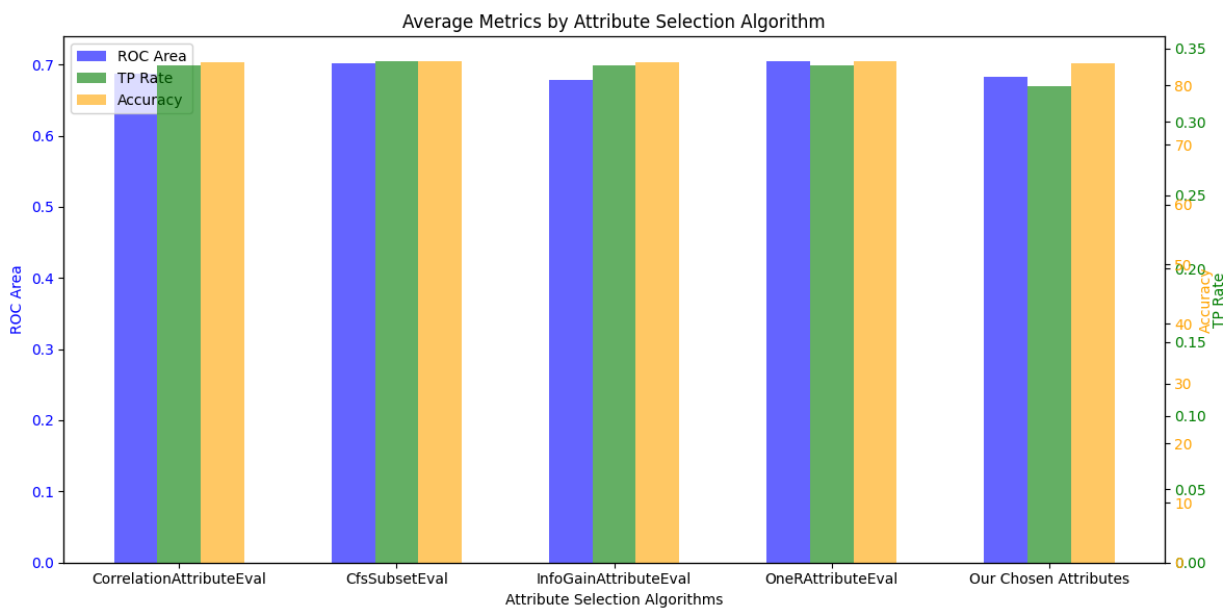
We decided to weight true positive the highest when comparing the results of the models, as it's very important that the model could accurately predict that named all-stars are all-stars.

Comparing all of the results, we determined the best model to be CfsSubsetEval with J48 Classification. It has the largest true positive rate of 0.684, an accuracy of 90.238%, and an ROC Area of 0.831. Although it does not make the top five in Accuracy and ROC Area, it's accuracy and ROC area are very close to the top 5, and having the highest true positive Rate proves its applicability.

Tied for second place are CfsSubsetEval with OneR Classification and InfoGainAttributeEval with OneR Classification, who have the highest accuracy, the second highest true positive rate, and the third highest ROC Areas.

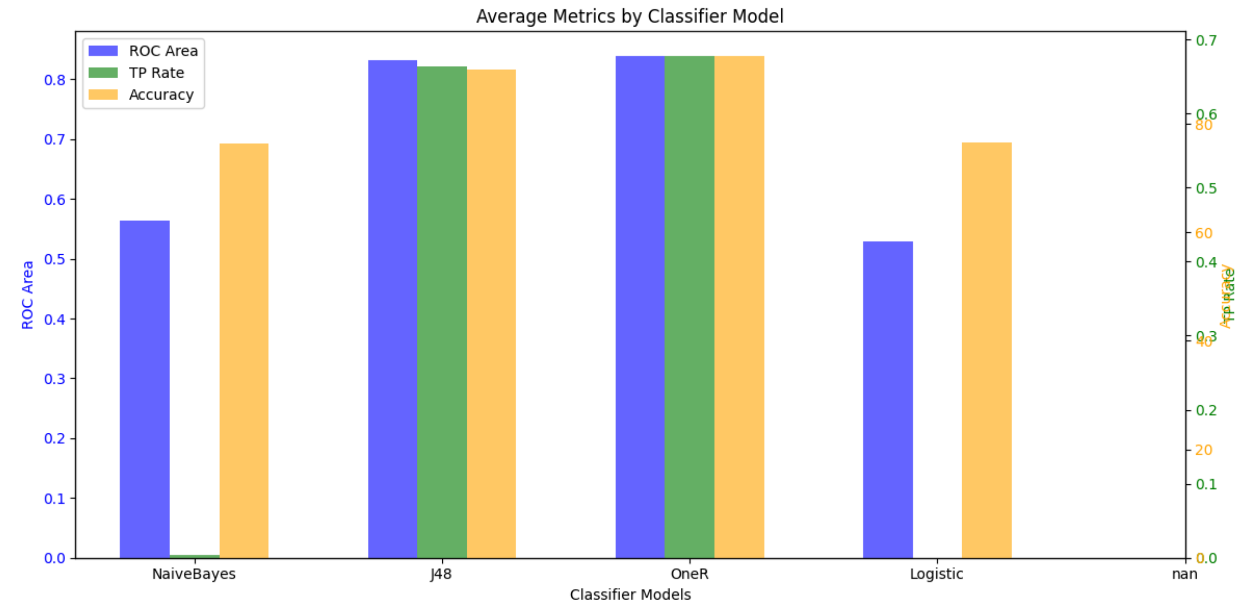
Part 8 - Interesting Trends

Examining the performance, we compared the average performance of the attribute selection algorithms to generalize trends. In the bar graph below, we can see that CfsSubsetEval performs the best on all metrics and OneRAttributeEval performs strongly as well. It was interesting to see that the attributes from the set we chose performed the worst, as from our basketball knowledge we assumed it would perform strongly.



We did the same for the classifier models, and it was clear to see that OneR resoundingly performed the best, with J48 performing strong as well. The NaiveBayes and Logistic models performed almost equally poorly on the data, and when examining their results we can see that both almost always classified every instance as being a non-allstar (0 TP rate).

The poor performance of NaiveBayes and the Logistic classification models can be attributed to their inability to handle the large class imbalance in the dataset. Both tend to default towards the majority class in cases of imbalanced data, as in this case, which attributed too their very low true positive identification. Additionally, both NaiveBayes, which assumes feature independence, and Logistic Regression, which relies on a linear decision boundary, may not capture the complex patterns necessary for correct predictions.



Another very important trend is that OneR, the best performing classifier model, selected Age or Wins as the rule attribute (Wins 3 times, Age twice). This shows that these are the two most useful attributes when determining if someone will be named an all-star. Logically, age makes sense, as strong players stay in the league longer, meaning that most likely an old player will be an all-star. A key example of this is Lebron James, who is 40 years old, has been in the league for 22 seasons, and is still being named an all-star. The Wins trend is a bit more difficult to explain, because all of the players who touched the court during the game get a credit for the win, meaning that all-stars would not necessarily have more wins in a season than the other starters on their team. More research is needed to determine the cause of wins being such a strong determinant of all-star selection. However, we have a theory that Wins is important due to fan base popularity. As we mentioned above fans have a huge part in selecting an all-star, we believe that the more wins a team has the more popular the team is. Therefore, the player on the winning team is more likely to get selected as an all-star.

Other notable trends were that although J48 did not lead in accuracy, it showed a strong performance in TP rates across multiple attribute evaluation methods, and that models like OneR and J48 maintained a high ROC Area across multiple attribute selection techniques.

Part 9 - Contribution of Team Members

The work was done collaboratively every step of the way either working on call with each other or delegating tasks and splitting up workload evenly amongst the group members.

Generally, Gus preprocessed the data and created the models in Weka, while Lalit synthesized that information and created the slideshow.

Part 10 - Data/Sources

Link for initial data: nba_player_stats_with_all_star_selection

Link for preprocessed data: preprocessed_nba_player_stats_with_all_star_selection

Link for train/test/validation splits: Train/Test/Validation Splits

We received our data from stats.nba and basketball-reference.