Social Media Power:
Strategic Targeting for NBA Player Sponsorship

Guy Dotan - Stats 404 Final Project

March 19, 2019

1 Setting the Stage

Sponsorships are as deeply integrated into the fabric of the sports landscape as the very players and competition itself. Every year around the world, billions of dollars are being invested into professional athletes for brand promotion. Sports stars are some of the strongest marketing influencers of any celebrity, but over the past few years trends have shifted dramatically in this industry. All of the world's biggest sporting competitions (NFL, NBA, Premier League, etc.) are reporting declining numbers of traditional TV viewers. That said, the sports sponsorship market remains as strong as ever due to the explosion in social media engagement.

With the focus of sports branding shifting toward social media, marketers need to focus their attention on these platforms. Of all the major sports leagues in America, the NBA is growing the fastest and is the most progressive at integrating and promoting social media. As shown below the top three yearly endorsement contracts in the NBA are valued at over $35 million (per Kaggle's social power dataset). In addition to the contractional endorsements, NBA stars are able to generate brand exposure organically through social media. Using proprietary computer vision technology, GumGum Sports, is able to track brand exposures across social media and attribute a corresponding media value. So far in the 2018-19 NBA season, LeBron James leads all players by generating almost $5.0 million of organic sponsorship content through Twitter, Instagram, and Facebook.

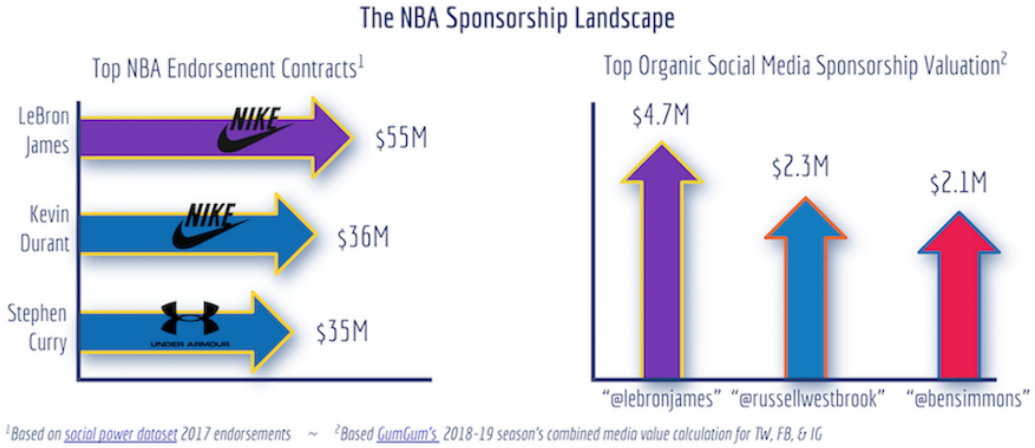


Figure 1

The relationship between the NBA and Twitter continues to grow as does the value brought in by that platform. With all of this data surrounding the power in valuing the NBA and social media, an important question emerges. **How should a brand/agency strategically determine the best players to target for sponsorships?**

2 The Data

In order to build a model that can identify NBA players with high social media value potential I pulled a dataset from Kaggle regarding the 2016-17 NBA season. The data includes a variety of on-the-court basketball statistics along with some social media metrics. Included were about 40 variables. The relevant variables in our dataset are grouped into the following categories:

Potential Input Variables:

- 26 basic statistics (PPG, 3PM, FG%, etc.)
- 6 advanced statistics (ORPM, DRPM, PIE, etc.)
- 6 other metrics (Age, Position, Wikipedia Page Views, etc.)

Potential Output Variables:

- Twitter Retweet Count
- Twitter Favorite Count

Before we begin to handle any missing values and potential issues with the dataset the goal is to determine on a high-level how we want to structure this model. For the target output variable we determined that **Twitter retweets** was the better metric to reflect social media influence. Retweets show a much more active engagement with media content than favoriting does and therefore would generate more brand value. In addition retweeting would spread that content to that user's network of followers which increases the original exposure even further.

As for the input variables, the goal of this exercise would be to develop a model that could use widely available data (basketball stats) to predict a more difficult to aquire and valuable metric as social media engagement. Therefore, we will focus our attention on the numerical basketball and team statistics. Right off the bat this eliminates position, player name, team name, and wikipedia page views.

3 Data Processing

3.1 Null values

First step toward handling our data was to resolve issues with null values. As shown below, only four columns had any null values. In cases where 3PT% and FT% were null, this was due to the player having taken no FT or 3PT attempts. As a result, we set these values equal to 0. In cases where Twitter Retweet and Favorite Count were null, we unfortunately had to remove those players as we needed this value to be our output metric.

```
3P%                       7
FT%                       2
TWITTER_FAVORITE_COUNT    3
TWITTER_RETWEET_COUNT     3
dtype: int64
```

PLAYER	3P	3PA	3P%	FT	FTA	FT%	TWIT_FAV_CT	TWIT_RETWEET_CT
Tyson Chandler	0	0	null	1.9	2.6	0.734	29	9
David Lee	0	0	null	1	1.4	0.708	56.5	35
Ian Mahinmi	0	0	null	1.4	2.4	0.573	10	5
Anthony Brown	0.6	2.5	0.259	0	0	null	3	3
Dragan Bender	0.7	2.3	0.277	0.1	0.3	0.364	null	null
Omer Asik	0	0	null	0.7	1.3	0.59	null	null
Miles Plumlee	0	0	null	0.6	0.9	0.641	11	3
Rakeem Christmas	0	0	null	0.7	1	0.724	13	3
Cole Aldrich	0	0	null	0.2	0.4	0.682	22	9
Bruno Caboclo	0.2	0.7	0.333	0	0	null	11	8
Alonzo Gee	0	0.2	0	0.4	0.7	0.556	null	null

3.2 Binning retweets

The second step needed in our data processing was to bin our output variable into larger buckets. For the purposes of our model it is not necessary to predict exactly how many retweets a player would have. Instead we just need to know the general range.

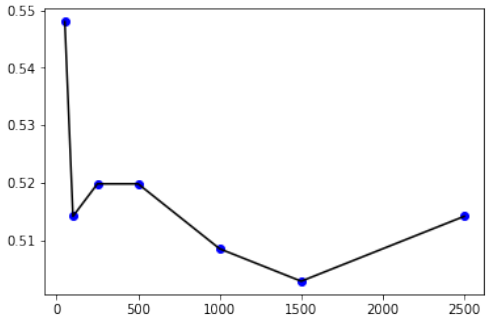
```
Out[185]: 10-50     95
          <10       82
          50-150    34
          150+      25
          Name: TWEET_CAT, dtype: int64
```

4 Modeling Approach

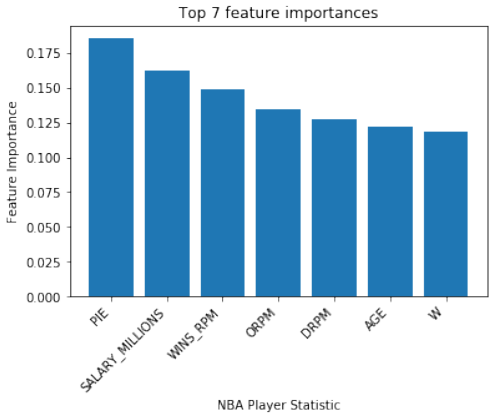
Before we begin to model our dataset, there was something to note about our input variables. One notable intricacy about the dataset is many of the advanced basketball statistics are a combination of the basic statistics. As a result there would be much covariance between these features. As a result we decided to narrow the feature set to just seven key metrics.

With the features determined, we decided to utilize the random forest modeling technique to predict which players fall into which retweet count bucket. After splitting the data into a 75-25 train-test set split we ran a random forest with up to 2500 trees, using a minimum leaf sample of 5, and balanced weight sampling.

As shown below we saw that the optimal OOB error occured at 1500 trees. In addition our feature importance plot showcased PIE (Player Inpact Estimate) and Salary (millions) as the most prominent features.



(a) OOB Error Plot



(b) Feature Importance Plot

5 Key Findings

The last step with analyzing our random forest model was to take a look at the prediction success. When applying our model to our validation set of 59 NBA players, our model accurately predicted the retweet bucket in just under half those cases (~47%). The prediction results are displayed below in the confusion matrix.

	<10	10-50	50-150	150+
<10	12	1	2	9
10-50	1	1	4	0
50-150	0	3	3	3
150+	7	0	1	12

```
In [176]: accuracy_score(y_valid, y_pred_valid)

Out[176]: 0.4745762711864407
```

6 Use Cases

Building a model that can predict an NBA's player's social media presence could be an extremely powerful tool for a brand or sponsorship agency. There could be a lot of untapped value potential in emerging NBA players and being able to identify those players would be critical. The following would be just a few use cases on how to utilize this model's potential.

6.1 Targeting Rookies

- With all rookies up for grabs after declaring for NBA draft, brands must determine which ones target with somewhat limited of information.
- By plugging in the rookie's NCAA statistics, rookie salary, and drafted team a branding agency could predict the social media power of the next great NBA star.

6.2 Trades / Free Agency

- As one would expect, changing teams can significantly affect a player's social media presence.
- Joining a top tier team could mean performing on a better team that would play more nationally televised games and postseason games. But joining a lower tier team could make the player the focal point of the franchise and also increase value.
- Also, joining a team in a bigger markets would obviously lead to increased exposure as well.

6.3 New Contracts

- Our model indicated that player salary was a top feature in predicting social media power.
- Targeting NBA players with a big pay raise in the offseason gives a major indication into the increase in social media value

7 Future Research

7.1 Instagram and Facebook

Although Twitter is the most active of the three social media platforms among NBA players, it is mainly a text-driven platform. Instagram and Facebook generate far more visual media (images and videos) and therfore would provide far more opportunities for product placement. For a more complete analysis of the power of social media sponsorships, both Instagrm and Facebook must be included.

7.2 Attendance

We know that the team plays a major role in a player's marketing power. There are many ways to attempt to quantify a team's impact. One of the most transparent methods would be to incoporate team attendence into the model. Attendence indicates not only how many eyes are at the game but also directly correlates with the size of the TV and social media market.

7.3 Season-to-Season Trends

This model was weakened by the fact that we are only working with a single season's worth of data. In order to build a more robust model, increasing the size of the dataset would be crucial. Not only would having more data points increase the quality of the model but it would also unable use to examine player trends. If an NBA player's stats seems to be improving year over year then we would assume his social media precense to follow a similar positive trajectory.

7.4 Postseason performance

Finally, our dataset was limited to regular season statistics. The NBA playoff brings in millions of dollars every season just through broadcasting rights. NBA players who perform in the playoffs and perform well experience vast increases in their exposure. An analysis of the branding value of an NBA player would not be complete without examining their playoff metrics as well.

8 Appdendix

8.1 Data Loading and Processing code

```
In [165]: import pandas as pd
          import numpy as np
          import random
          from sklearn.model_selection import train_test_split
          from sklearn.ensemble import RandomForestClassifier
          import matplotlib.pyplot as plt
          from sklearn.metrics import accuracy_score, confusion_matrix

          PATH = ('https://s3-us-west-1.amazonaws.com/uclastats404-project/'
                      'nba_2017_players_with_salary_wiki_twitter.csv')
          DF = pd.read_csv(PATH)

In [151]: null_columns=DF.columns[DF.isnull().any()]
          print(DF[null_columns].isnull().sum())
          df_subset = DF[["PLAYER","3P","3PA","3P%","FT","FTA","FT%",
                          "TWITTER_FAVORITE_COUNT","TWITTER_RETWEET_COUNT"]]
          display(df_subset[DF.isnull().any(axis=1)])
          nba = DF[np.isfinite(DF['TWITTER_RETWEET_COUNT'])]
```

8.2 Binning Code

```
In [185]: def bin_retweet_count(retweets: float) -> str:
              if retweets < 10:
                  retweet_cat_group = "<10"
              elif (retweets >= 10) & (retweets < 50):
                  retweet_cat_group = "10-50"
              elif (retweets >= 50) & (retweets < 150):
                  retweet_cat_group = "50-150"
              else:
                  retweet_cat_group = "150+"
              return retweet_cat_group

          # prevent pandas warning message for SettingWithCopyWarning
          # taken from https://stackoverflow.com/questions/20625582/
              # how-to-deal-with-settingwithcopywarning-in-pandas
          pd.options.mode.chained_assignment = None
          nba['TWEET_CAT'] = nba['TWITTER_RETWEET_COUNT'].apply(lambda
                                                                x: bin_retweet_count(x))
          nba['TWEET_CAT'].value_counts()
```

8.3 Random Forest Code

```
In [179]: df_train, df_valid = train_test_split(nba,
                                                test_size=0.25,
                                                random_state=2019,
                                                stratify=nba['TWEET_CAT'])

          y = df_train['TWEET_CAT']
          X = df_train[['AGE','WINS_RPM','SALARY_MILLIONS','W', 'ORPM', 'DRPM','PIE']]

          ### --- Step 1: Specify different number of trees in forest, to determine
          ###             how many to use based on leveling-off of OOB error:
          n_trees = [50, 100, 250, 500, 1000, 1500, 2500]

          ### --- Step 2: Create dictionary to save-off each estimated RF model:
          rf_dict = dict.fromkeys(n_trees)

          for num in n_trees:
              ### --- Step 3: Specify RF model to estimate:
              rf = RandomForestClassifier(n_estimators=num,
                                          min_samples_leaf=5,
                                          oob_score=True,
                                          random_state=2019,
                                          class_weight='balanced',
                                          verbose=0)

              ### --- Step 4: Estimate RF model and save estimated model:
              rf.fit(X, y)
              rf_dict[num] = rf

          # Compute OOB error per
          # https://scikit-learn.org/stable/auto_examples/ensemble/plot_ensemble_oob.html
          oob_error_list = [None] * len(n_trees)

          # Find OOB error for each forest size:
          for i in range(len(n_trees)):
              oob_error_list[i] = 1 - rf_dict[n_trees[i]].oob_score_
          else:
              # Visulaize result:
              plt.plot(n_trees, oob_error_list, 'bo',
                       n_trees, oob_error_list, 'k')
```

8.4 Feature Importance code

```
In [181]: # Feature importance plot, modified from:
          # https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html
          top_num = 7
          forest = rf_dict[2500]
```

```
importances = forest.feature_importances_

# Sort in decreasing order:
indices = np.argsort(importances)[::-1]
xvarlist = X.columns[indices]

# Plot the feature importances of the forest
ax = plt.gca()
plt.title(f"Top {top_num} feature importances")
plt.bar(range(top_num), importances[indices[0:top_num]])
plt.xticks(range(top_num))
ax.set_xticklabels(xvarlist, rotation = 45, ha='right')
ax.set_xlabel("NBA Player Statistic")
ax.set_ylabel("Feature Importance")
plt.show()
```

8.5 Confusion Matrix code

```
In [182]: y_valid = df_valid['TWEET_CAT']
          X_valid = df_valid[['AGE','WINS_RPM','SALARY_MILLIONS','W', 'ORPM', 'DRPM','PIE']]

          y_pred_valid = forest.predict(X_valid)

          conf_mat_valid = confusion_matrix(y_true=y_valid,
                                            y_pred=y_pred_valid)

          tweetlabels = ["<10", "10-50", "50-150", "150+"]
          class_names = tweetlabels
          conf_df = pd.DataFrame(conf_mat_valid, class_names, class_names)
          display(conf_df)
```