Using Machine Learning to Capitalize on Mispriced Odds Through Underdog Betting in the NBA

Matthew Agard, Pranab Islam, Jack Jones, Sanjay Subramanian

I. BUSINESS UNDERSTANDING

Since the strike-down of a decades-old federal ban on sports betting in the United States by the Supreme Court in 2018 [1], the world of online sports gambling has exploded, with betting operators such as DraftKings and FanDuel hosting sportsbook options in a number of states. This has opened the door to a new wave of virtual users, providing access to all major sports leagues such as the NBA and NFL, and even not so well-known entities such as the Swedish Handball Federation. Sports betting, rightfully so, is known to be a risky proposition. However, building a model that is able to consistently exploit mispriced bookie odds with high confidence can lead to a safe and positive return on investment (ROI) in the long run, even while accounting for large losses. Sportsbooks generally have to consolidate bet line influx from zealous bettors on top of considering built-in game outcome probabilities. Sportsbooks set betting lines with the intent of maximizing their chance to profit, usually by adjusting lines as money comes in on a particular side of a bet. Dictating this line movement is an attempt to hedge exposure and maximize profits [2]. Naturally, a majority of bets will be placed on the favorite. This can result in improved odds for the longshot as the line shifts to attract bettors to the underdog. Despite this, underdog betting remains a mostly unexplored market, with most strategies involving safe bets on low-return favorites, leading to a slow, long-term profit accumulation at best. In the NBA, approximately one-third of all matchups are won by the underdog, indicating a huge potential for positive ROI on underdog betting. The precision threshold required for netting profit is not high given the expected value return (high risk-high reward) on underdogs in pre-game money lines. Presented with this supervised classification problem, there is an opportunity to use machine learning to take the ‘high’ out of ‘high-risk’ and identify instances in which we could calculate the probability of an underdog winning a game at a more confident level than the current money lines would suggest. Using a variety of rolling counting statistics and other performance indicators from 12 seasons-worth of NBA matchup data, we attempt to navigate and pinpoint these idiosyncrasies which allow underdogs to win at a non-trivial rate. Further focusing on matchups which exceed a specific favorite-underdog disparity threshold (2:1 or greater) allows us to specify those matchups in which a correct prediction will result in a profit in excess of double the initial bet. We simulate a betting portfolio on the most recent season of the NBA using a variety of models compared against a baseline strategy of random guessing, a technique which most likely captures the approach of modern-day casual bettors. A successful strategy, unburdened by the restraints faced by sportsbooks, is able to capitalize on mispriced odds and return high magnitude profits while also potentially being marketable as a premium service for budding bettors.

II. DATA UNDERSTANDING

A. *Data Collection & Feature Construction*

All data used in this project came exclusively from Bet Explorer[1], Basketball Reference (BREF)[2], and official NBA statistics trackers[3], and covered every NBA game from the start of the 2008 – 2009 season to the end of the 2019 – 2020 season. Courtesy of open-source NBA odds scraping code found on GitHub[4], we scraped Bet Explorer for dates, opening decimal odds, and closing decimal odds of the home and away teams associated with every game in a given NBA season. We debugged and repurposed the scraper to fit the scope of our project. For each game, we computed the mean of all provided bookie odds to represent a group consensus on how the game should be priced, a very common practice in market analysis from a financial perspective. We made use of another open-source GitHub repository[5] for almost all of our web scraping of BREF. Functions such as "get_schedule()" and "get_game_logs()" returned dataframes containing essential features such as game scores

[1] betexplorer.com/.
[2] basketball-reference.com/.
[3] nba.com/stats/leaders/.
[4] github.com/calebcheng00/nba_predictions/.
[5] github.com/vishaalagartha/basketball_reference_scraper

and team statistics for every game in a given season. We rewrote a portion of the API to account for a number of bugs, and subsequently constructed tailored features based on phenomena that could significantly impact the outcome of an NBA matchup. These features include concepts such as number of impact players involved in a game as well as presence of all-star caliber players on either roster. From the official NBA statistics website, we scraped team play style statistics – these are quantities that describe how a team plays relative to other teams based on certain generalizable categories considered important to basketball strategy. These included percentile rankings based on categories such as transition points, cut defense, pick-and-roll offense, etc.

B. *Selection Bias & Concept Drift*

We define a data instance as a single NBA matchup where one of the teams is considered a heavy underdog. We determine the presence of a heavy underdog by evaluating whether the American odds for either team has absolute value greater than or equal to 200 (indicating 2:1 odds for the underdog). Our data includes every NBA game from the beginning of the 2008 – 2009 season through the end of the 2019 – 2020 season, all of which had public betting availability. We consider that our data is more or less the entire population of games relevant to our business case. While this is not technically true (basketball and betting were not invented in 2008), evolution of the game and a dynamic shift in strategies indicate potential concept drift [3]. In addition, the betting market (as well as betting data) from over a decade ago was less liquid and institutionally robust as it is now; retrieving data from much earlier than 2008 risks incorporation of data that does not align with current betting market and game tendencies [4]. While our data may still exhibit some degree of concept drift due to certain key events reshaping the way basketball is played (for example: Stephen Curry's 3 point eruption [5] and James Harden's elite rule bending [6]), we attempt to control for this by treating the starting year of each selected training dataset as a hyperparameter to tune. There exists intentional selection bias in our data due to the filtration for games involving only heavy underdogs, which we view as a reasonable adjustment in order to obtain model sensitivity towards these types of relevant matchups.

C. *Exploratory Data Analysis*

Initially, statistical features were classified by “Home Team” and “Away Team”. After checking each feature’s mutual information with our target variable alongside a random forest to examine feature importance, we were able to drop all play style statistics (which made up almost half of all features), as they lacked any substantial information relative to other features. This was likely due to the nature of these statistics being cumulative averages from the prior season, leading to a lack of both notable information about a current season and ability to be transformed into a rolling average. We analyzed the collinearity of our data and concluded that many counting statistics, if not already a linear combination of another group of statistics, are computed using a similar set of metrics. We eliminated obvious linearity (for example, amongst 3-point attempts, 3-points made, and 3-point percentage, there exists a redundant feature) and left other features untouched in order to provide our models with all available information and let them handle further dimensionality reduction.

D. *Leakage*

An initial inspection of all statistics scraped via the BREF API revealed the need to transform counting statistics into rolling averages in order to avoid including game outcome information that would be unavailable to a prospective model at the time of betting. In order to eliminate leakage on this front, we created dynamic rolling average windows ranging from five games to fifty games and utilized these windows during the parameter tuning phases of modeling. There were notable leakage issues uncovered in the modeling phase which required further calibration.

III. DATA PREPARATION

A. *Data Integration*

Following the construction of dataframes containing our features of interest, we began the process of merging these features into a single, fully integrated dataframe. Initially, we performed an inner join on our dataframes using a unique game ID ([“Game Date”, “Home Team”, “Away Team”] composite key), but we quickly discovered significant data loss transpiring due to incorrect dates listed in our bookie odds dataframe – these dates were often either a day ahead or day behind the true game dates as indicated in the BREF dataframes. We combatted this by defining temporary unique game ID indicator columns representing plus-or-minus one day for each game in the bookie odds dataframe – we performed inner joins between this updated bookie odds dataframe and our BREF dataframe in order to consolidate unique games under a single, accurate date and produced our fully integrated dataframe. Lastly, we filtered our dataframe to keep only the games with heavy underdogs based on closing odds. This required that at least one team have decimal odds in the range $(1, 1.5] \cup [3, \infty)$, which respectively equate to $\leq -200 \; and \; \geq +200$ in American odds.

B. *Target Variable*

Our business case of predicting underdog win probability naturally lends itself to a binary outcome target variable ("Underdog Win"), with values of 0 and 1 representing an underdog loss or win respectively. Figure 1 shows the distribution of our target variable across all data instances that met our odds filter criteria. From 2008 – 2020, an NBA underdog had an expected win probability of approximately 0.23. This distribution can be further broken down by season, as seen in Figure 2. Our target variable was not present in any features scraped from Bet Explorer or BREF – we constructed it by defining intermediary indicator variables whose values we used to determine whether an underdog won or lost a given game.

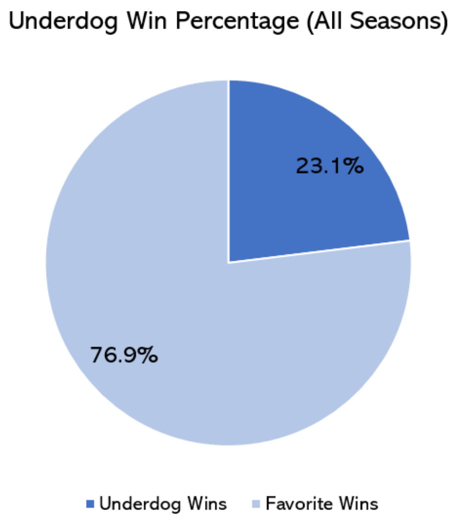

Figure 1

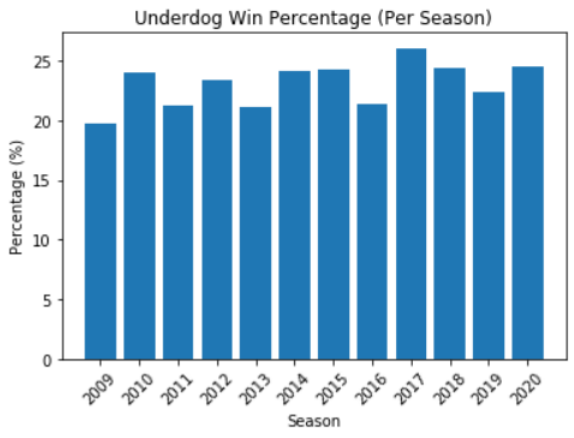

Figure 2

C. *Domain-Based Feature Engineering*

The NBA is a league driven in part by matchups between individual players, particularly high-impact players such as all-stars. We engineered three distinct features to capture this phenomenon. The first was the number of projected NBA all-stars each team possessed. This was accomplished by selecting players in the 95th percentile or higher for the advanced stat value over replacement player (VORP) – which measures how big of a contribution a given player makes to their team compared to a replacement-level player – and aggregating the count of said players by team. Other starters and key role players play a significant role in team success as well. This led us to implement a feature which indicates the proportion of impact players available to a given team. We identified the top seven players on each team based on minutes played and computed the proportion of these players that were unavailable due to injury, suspension, etc. at a particular point in the season using injury reports from BREF. Lastly, we computed the median roster age for each team to include potential cues regarding the effects of experience and veteran presence on game outcome. In addition to individual matchups, it is critical to quantify team matchups, so we engineered three team features: a team's prior wins vs. a given opponent, rolling win percentage, and cumulative win percentage. The first of these features aims to quantify head to head matchups between teams, which may hold more predictive power than merely observing individual records. The latter two features attempt to quantify whether a team is on a "hot streak" and its cumulative overall performance. As with counting statistics, we calculated rolling win percentage using a window ranging from five games up to fifty games. The first game of the season for each team was removed to eliminate null values from our data.

IV. MODELING AND EVALUATION

A. *Evaluation Metrics*

Common performance metrics for classification problems include accuracy, AUC, precision, recall, and log-loss. In our case, dealing with class imbalance, accuracy does not provide any meaningful measure of model performance. A betting model with high levels of confidence in its predictions is most desirable – to this end, errors in classification probabilities as opposed to purely binary classification scores are important to optimize in order to ensure the model is able to attain a reasonably high level of confidence in its predictions. This led us to prioritize minimization of log-loss – which considers underlying probabilities – in order to reduce model uncertainty. Also important to model success are precision and recall, as these metrics more directly determine profit potential. In the case of sports betting, false positives are much more detrimental than false negatives, as they result in monetary loss – optimizing for precision allows the model to develop stricter penalties for false positives, while optimizing for recall ensures a realistic amount of positive-class predictions are made. Accounting for both precision and recall through their harmonic mean (F1-score) allows us to find the potential "sweet spot". The global success criteria, no matter the evaluation metric used, will be ROI, as any successful betting model should be able to make money.

B. *Feature Engineering*

Initial mutual information analysis prior to the modeling stage indicated lack of correlation between all of the team play style statistics and game outcome. As these statistics were not rolling and thus did not update each game along with the counting statistics, omitting them from our model was reasonable. Another aspect of our raw datasets was the stratification of statistics into "Home" and "Away". Although each variation of these two labels are connected in a game between two teams, a model cannot distinguish this paired relationship. In order to alleviate the lost sensitivity, we re-engineered the features to the format "Underdog" *minus* "Favorite" to capture the relative difference in game performance between either side (outputting positive values if an underdog had outperformed their opponent in a certain statistic over a rolling window of time). This served to not only condense the number of features by almost half, but also retained all statistical information and sensitized it to each matchup. Initial modeling on both versions of training data confirmed these sentiments. Additionally, initial modeling yielded surprisingly positive results, which led us to critically reexamine included features for leakage. We determined that the methodology employed in constructing the team all-stars statistic resulted in leakage due to the fact that VORP values being used were reflective of end-of-season performance. Additionally, minutes played in specific games were being used to determine whether or not impact players were available in that game. We removed these features during the modeling stages to eliminate this potential leakage. One important aspect to preparing our data for modeling was standardization of our features in order to eliminate bias due to the varying scales and magnitudes of our statistics. Principal component analysis was also employed in order to account for any remaining collinearity, while providing the added benefit of significantly reducing model runtime. An attempt to incorporate polynomial features to capture nonlinear dependencies in our dataset was prohibited by excessive runtimes.

C. *Model Selection*

In selecting a baseline, we considered the modeling capabilities of an average bettor, and likened this to a system of random guessing, whereby "gut-feeling" founded on moderate knowledge of underdog success rates could drive strategy. Given the proportion of heavy underdog wins (0.23) over the range of years from 2008 to 2018, a random guessing model simulated 10000 times on the 2018 – 2019 and 2019 – 2020 seasons achieved an average ROI of -3.27%. The average precision and log-loss for this model were 0.245 and 0.539 respectively, indicating that any further fine-tuned model should at least be able to outperform these scores. For binary classification problems, logistic regression has been proven to be robust and reliable given small to moderately sized training data. An initial model with no feature scaling, dimensionality reduction, or hyperparameter tuning trained on an arbitrarily chosen training set of games from 2014 – 2018 with a rolling average window of 5 games returned a 0.667 precision with a 0.993 ROI when tested on the 2018 – 2019 and 2019 – 2020 season. At first glance, the problem had seemingly been solved, but a recall score of 0.0066 and log-loss of 0.541 indicated both a lack of confidence and elevated uncertainty as compared to even the baseline of random guessing. With only 3 positive predictions in a test set of 1288 games (301 of which were positives), it turned out there was very little signal being picked up from the data in its initial form, and very limited business potential from such a small sample of predictions, none of which exceeded a probability score of 0.55. In order to construct a framework by which to evaluate the efficacy of our ensuing models, we subjected each to an iterative grid search process with 5-fold cross validation while tuning a number of model-specific parameters such as regularization weight, class weight, kernel, max depth, etc., along with global hyperparameters such as training set size (ranging to include data as far back as 2009 or as recent as 2016) and rolling average window (ranging from five to fifty games). In addition, a number of scoring parameters were prioritized across each iteration, including precision, recall, F1-score, and log-loss.

i. Logistic Regression

Initial modeling stages, similar to the baseline logistic regression, showed many iterations with surprisingly high precisions and ROIs. A revisiting of features and subsequent removal of those we suspected had contributed to leakage led to a more realistic set of results, which were then improved by the reformatting of statistics features into relative differences. However, an evaluation of our metrics indicated a high variance in model performance. Aside from marked improvement in performance with the balancing of classes, we observed no other consistent trends in terms of parameter optimization. While certain models presented positive ROIs (up to 5.5%) with high recalls and moderately low precisions (0.25 – 0.28), multiple nearly equivalent models trained on the same data in each case outputted negative ROIs. These observations, in addition to log-losses consistently above 0.6, led us to believe the model may have occasionally been getting lucky, as the average ROI across the best versions of all training size and rolling window average iterations was negative.

ii. Support Vector Machine

We fit various support vector machine (SVM) models to explore potential non-linear class boundaries. SVMs contain implicit, built-in kernels that enlarge the feature space to accommodate a non-linear hyperplane between classes. This kernel approach is more efficient than logistic regression in stratifying non-linear properties of data. We employed both linear and radial basis function (RBF) kernels. In order to use the RBF method to explore a non-linear class boundary, we hyper-parametrized over a range of gamma values. Given the imbalance in our data, we attempted to apply Synthetic Minority Over-Sampling Technique (SMOTE) to up-sample the minority class in our target variable. A model trained on the dataset with natural default imbalance performed better than a SMOTE-balanced dataset. The best nontrivial model (trivial models predicted all underdogs to lose, thus returning a precision, recall, and ROI of 0.0) had a linear kernel and default class weighting, producing an ROI of 7.51% with a precision of 0.274

iii. Tree-based Methods

With the understanding that more "simple" models like the logistic regression or support vector machines may not possess the sensitivity needed to capture the signal of interest, we also decided to utilize both gradient boosted trees and random forests. While tree-based methods are more sensitive to hyperparameter tuning, their inherently nondeterministic nature makes them harder to interpret. As in the case of logistic regression, initial modeling performed respectably with minimal parameter tuning. We utilized multiple virtual instances on Google Console and ran parallelized grid searches to find optimal parameter combinations. A few models generated ROIs close to 30%, with consistent expected value positive decisions; this was a cause for skepticism, since we generally assume that any reasonably liquid market is efficient. After discarding previously discussed problematic features, we repeated the grid search process. Using log-loss as our evaluation metric of interest in order to enhance the confidence of predictions, we consistently observed trivial models. This pointed to a potential lack of signal in the data available to our models, and thus an inability to identify winning scenarios for underdogs. Understanding the limitations of our dataset and the true end goal of our project – maximizing ROI – we shifted our focus from heightened model confidence to a capitalization of acceptable opportunities. Due to the filtering of the data to only include games with heavy underdogs, the precision required for a model to successfully return positive ROI lay somewhere around 0.3 (each correctly guessed game produces an individual ROI of at least 200%). To this end, we optimized our models for F1-score in order to augment precision while keeping recall high enough to stabilize long-term ROI. The log-losses of both models were greater than the log-loss in the random guessing model. However, ROI was consistently higher due to what we describe as the ability of our models to be "smart coin flippers"; that is, they are able to capture signal that is strong enough to consistently turn a profit, even if confidence is not high. The top iterations of random forests and gradient boosted trees outputted precisions of 0.3 and 0.42 with recalls of 0.36 and 0.05, respectively, when tested on the 2019 – 2020 season. For the random forest model, an average of the top quartile's prediction probabilities vectors resulted in an ROI of 3.9%. We achieve a 101.8% ROI for the gradient boosted tree due to its relatively high precision; a limited number of bets, evidenced by low recall, likely boosts ROI as the model need only make a few "smart" guesses to double its initial investment. Alternatively, perhaps the model has found an interesting pattern in a small subset of games which could potentially be extrapolated to a variety of out-of-sample datasets given the right cues. In Figures 3 and 4, we note the precision-recall relationship in these models. Whereas the random forest model seems to maintain stability in its precision as recall increases, the gradient boosted tree follows a more standard downward trend.

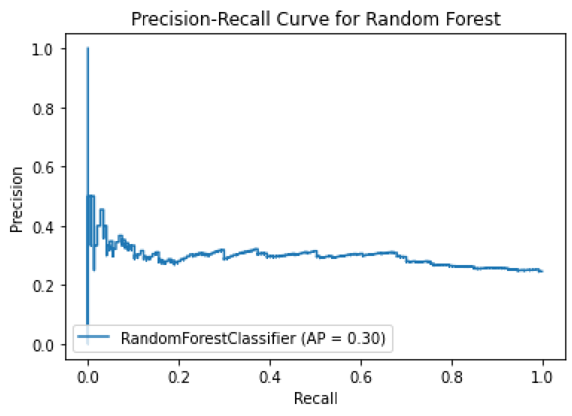

Figure 3

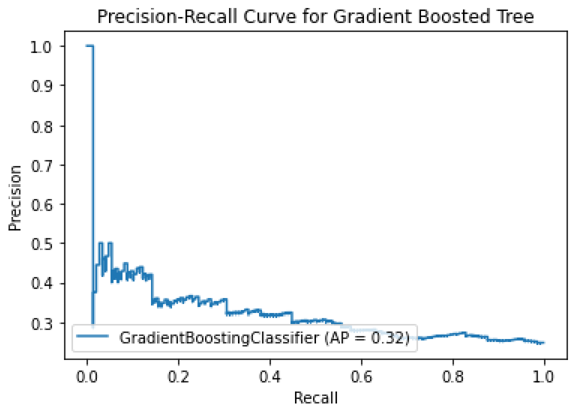

Figure 4

D. *Betting Portfolio*

In order to simulate the returns of our model compared against the baseline strategy of random guessing, we constructed a betting portfolio using the 2019 – 2020 season as a test set. For each of the random guessing, random forest, and gradient boosted tree models, we show ROI as a function of the number of games the model bet on as the season progressed (shown in Figure 5). We see that the gradient boosted tree model produces a large ROI over a small subset of games, while the random forest model stabilizes earnings through occasional considerable wins over a longer period of time. A random guessing approach, simulated 10,000 times, bets on close to one-sixth of games in the training set with negative returns. It is important to note that given different variations of the test set, model performance may be drastically different.

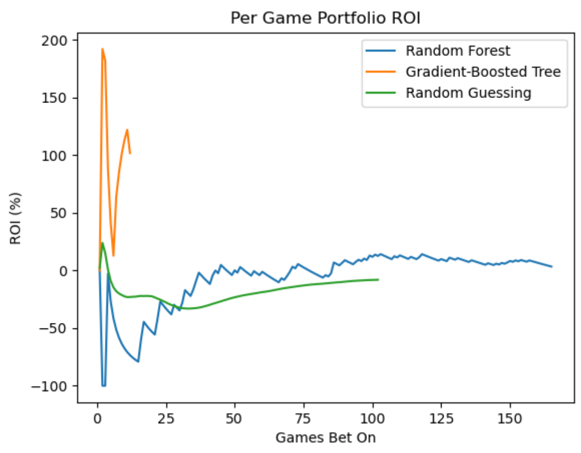

Figure 5

Random Forest ROI	3.9%
Gradient Boosted Tree ROI	101.8%
Random Guessing ROI	-8.5%

Table 1

A more comprehensive reporting of metrics for each model can be found in Table A1 of the Appendix.

V. DEPLOYMENT

A. *Infrastructure*

Our model relies on real time prediction capability hours before a slate of games, so it is imperative to integrate with a high-functioning production pipeline which scrapes desired game statistics and betting odds and outputs predictions quickly and efficiently. In doing so, we will be able to construct a cohesive betting portfolio which capitalizes on mispriced odds and maximizes ROI. As a monitoring strategy, if a given model gives us ample reason to reject the null hypothesis that it is no better than random guessing, we can decide to invest capital and oversee returns. Poor decision-making can be weeded out and/or minimized over time, especially as our resources improve. Given the budding sports betting markets, our portfolio has the potential to entice avid amateur bettors who seek an advantage over bookies. The target market exists to offer a subscription service. whereby bettors pay a fee to access tout sheets. Such revenue could be invested back into model optimization.

B. *Ethical Considerations*

Over $20 billion has been bet through US-based sportsbooks, and eighteen states possess regulated sports betting markets. However, of these states, half have not allocated any funds to problem gambling services [7]. Potential harm caused by gambling addiction is a major ethical concern to consider in deployment. Although our model is rooted in statistical theory and trained over thousands of game logs, participating bettors enter into an agreement at their own risk [8]. Given our potential to profit off the betting market, there exists a moral duty to dedicate a portion of revenue to problem gambling services which address the negative effects, stimuli, and consequences associated with gambling.

C. *Future Work*

In the future, we would look to incorporate rolling play style statistics into our predictor space. These metrics crucially underscore how a team matches up against opponents. Additionally, we would train our models using a comprehensive dataset without filtering for heavy underdogs in order to allow the models to self-identify these games without explicit user input. In general, betting on game spreads as opposed to money lines presents a more profitable market, as the number of games where the underdog outperforms expectations is higher. Lastly, there is an opportunity to extrapolate these models to other markets, especially college basketball, where there is both a lack of advanced statistics available to bookies and a large volume of games.

D. *Closing Remarks*

We have attempted to address a complex modeling problem in a niche subset of the betting market. Exploitation of a such an efficient market can lead to massive returns, but the ability to capture signal from available data is not always a successful endeavor. While we were unable to find highly confident models that could output positive ROI with consistently low variance and log-loss, we uncovered hints of potential signal and luck embedded within our models' prediction mechanisms, indicating potential for finetuning and further optimization. The realization of positive ROIs in our random forest and gradient boosted tree models with precisions hovering around 0.3 indicates some success over baseline strategies while boasting both a potential to profit in the long-term and the dream of one day beating Vegas at their own game.

REFERENCES

[1] Holden, John. “Prohibitive Failure: The Demise of the Ban on Sports Betting”. 23 Sep. 2018. Georgia State University Law Review, Forthcoming, Available at SSRN: https://ssrn.com/abstract=3253906

[2] Cortis, Dominic (2015). "Expected Values and variance in bookmaker payouts: A Theoretical Approach towards setting limits on odds". Journal of Prediction Markets. 1. 9.

[3] Marius. “The History of Shot Distribution in the NBA.” Medium. 28 Mar. 2020. medium.com/@7FS/the-history-of-shot-distribution-in-the-nba-a7e3107c45aa.

[4] Kirkgoldsberry. “How Mapping Shots In The NBA Changed It Forever.” FiveThirtyEight, 2 May 2019, fivethirtyeight.com/features/how-mapping-shots-in-the-nba-changed-it-forever/.

[5] Rob. “How Steph Curry Changed the World of NBA through Data Analytics.” DataTalent, www.datatalent.io/blog-post/how-steph-curry-changed-the-world-of-nba-through-data-analytics.

[6] djallen23, et al. “James Harden Motivates the NBA to Add Detail to the Rule Book.” Basketball Society. 24 Sep. 2019. basketballsocietyonline.com/james-harden-nba-rule-book-travel.

[7] Purdam. “Sports betting's growth in U.S. 'extraordinary'.” ESPN. 14 May 2020. https://www.espn.com/chalk/story/_/id/29174799/sports-betting-growth-us-extraordinary

[8] Killick, Elizabeth; Griffiths, Mark. 16 Apr. 2018. "In-Play Sports Betting: a Scoping Study". International Journal of Mental Health and Addiction. 17 (6): 1456–1495. doi:10.1007/s11469-018-9896-6.

CONTRIBUTIONS[6]

Matthew: Scraping and merging, feature construction, domain-based feature engineering, data preparation, betting portfolio

Jack: Feature construction, SVM hyperparameter tuning, deployment infrastructure, ethics, future considerations

Pranab: Scraping and merging, tree-based hyperparameter tuning, modeling and evaluation, betting portfolio

Sanjay: Business understanding, merging, construction of baseline, logistic regression hyperparameter tuning, modeling and evaluation, betting portfolio

[6] Our GitHub repo can be accessed at: github.com/sanjsub/DS-GA-1001-Odd-Sharks/

APPENDIX

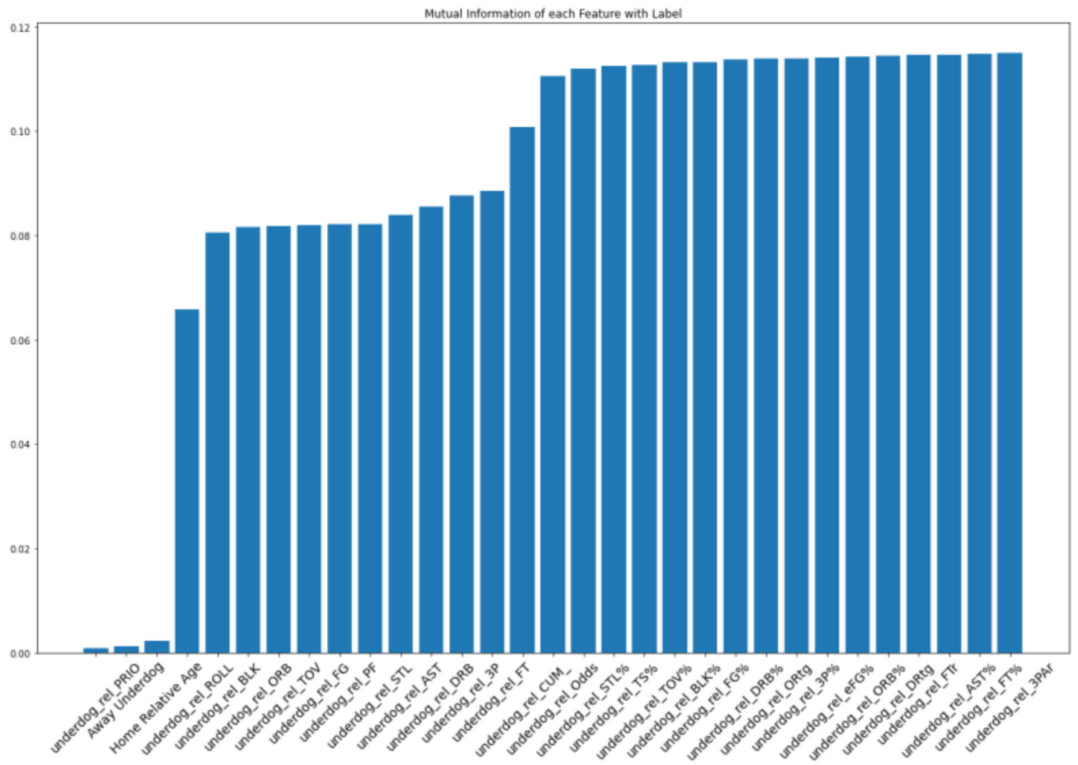

Figure A1

	Random Guessing[1]	**Logistic Regression**[2]	**Support Vector Machine**[3]	**Random Forest**[4]	**Gradient Boosted Trees**[5]	**Best**
Best Precision	0.28	0.29	0.27	0.32	0.55	Gradient Boosted Trees
Average Precision	**0.25**	**0.26**	**0.13**	**0.31**	**0.42**	**Gradient Boosted Trees**
Best Recall	0.88	0.74	0.71	0.37	0.07	Random Guessing
Average Recall	**0.77**	**0.64**	**0.26**	**0.36**	**0.05**	**Random Guessing**
Best AUC	--	0.59	0.58	0.59	0.61	Gradient Boosted Trees
Average AUC	--	**0.58**	**0.54**	**0.58**	**0.59**	**Gradient Boosted Trees**
Best Log-Loss	--	0.66	--	0.63	0.56	Gradient Boosted Trees
Average Log-Loss	**0.54**	**0.71**	--	**0.63**	**0.57**	**Random Guessing**
Best ROI	45%	5%	8%	8%	84%	Gradient Boosted Trees
Average ROI	**(3%)**	**(2%)**	**(0%)**	**6%**	**37%**	**Gradient Boosted Trees**

1. Random guessing shows a Monte Carlo simulation of Bernoulli(p) trials where p = 0.23 and whenever the random variable = 1, we bet on the game at hand for the 2019 - 2020 NBA season
2. Table shows the performance of eighteen of our best models trained with different hyper parameters
3. Table shows the performance of six of our best models trained with different hyper parameters
4. Table shows the performance of the five (out of twenty) best models trained with the same hyper parameters; variability comes from inherent randomness in random forest algorithm
5. Table shows the performance of the twenty best models trained with the same hyper parameters; variability comes from inherent randomness in the gradient boosted trees algorithm

Table A1