EXPERIMENT REPORT

Student Name	Ron Au
Project Name	NBA Career Prediction - Week 3
Date	2021-02-21
Deliverables	Au_Ron-week3_kaggle-nba.ipynb https://github.com/ronvoluted/kaggle-nba

1. EXPERIMENT BACKGROUND	
Provide information about the problem/project such as the scope, the overall objective, expectations. Lay down the goal of this experiment and what are the insights, answers you want to gain or level of performance you are expecting to reach.	
1.a. Business Objective	Using a dataset of 8,000 NBA rookies and 19 features, train a model to predict their probability of having an NBA career longer than 5 years. The application of this problem is a Kaggle competition where a list of 3,799 unlabelled samples need to have their >5Yr probabilities predicted and recorded on a range from 0 to 1. The objective is to use the competitive environment to practice improving our data science methodology and assessing how different techniques are more, or less relevant to specific training problems. By operating in teams, it will also promote the use of good code sharing practices.
1.b. Hypothesis	Previous experiment revealed these observations: ● 4,954 of 8,000 rows contained errors where values were negative ○ Dropping these rows always resulted in worse results ○ “It is possible that imputing the values or engineering new columns, instead of dropping them, may result in improved performance.” ● 5 to 1 imbalance in binary classifications. Various model and hyperparameter combinations resulted in: ○ 15 - 18% false negative rate ○ 7% average true negative rate ○ <1% false positive rate, often 0 ● Despite expectations based on visualisation, “A logistic regressor is quite viable for this data set.” ● Negative values should: ○ be imputed instead of having their rows dropped ○ be imputed via features instead of the absolute of the value It is hypothesised that engineering features using basketball domain knowledge will yield improved results. Features that emulate NBA analysis ratings will be attempted and tested. **Hypotheses for data preparation and feature engineering**

	A series of 16 different data preparation and feature engineering options will be tracked using MLflow Tracking. Each one is based on a hypothesis on why it might result in better scores. The options and their reasoning are below: Data preparation options ● **Impute negative values:** Fix definite errors ● **Upsample with smote:** Balanced dataset ● **Downsample randomly:** Balanced dataset ● **Recalculate percentages:** Fix definite errors Domain knowledge feature engineering options ● **Add Possessions:** Approximate metric of time spent with ball ● **Add Points per Possessions:** Metric of scoring ability when given the opportunity VS simply points per game ● **Add 3-pointers per 100 Possessions:** As above ● **Add Field Goals per 100 Possessions:** As above ● **Add Free Throws per Games:** Metric of how often player is chosen to free throw, not just their free throw accuracy. Suggests team confidence in player ● **Add 3-Pointer % > 75% of mean:** Metric of being better than below average ● **Add Field Goal % > 75% of mean:** As above ● **Add Free Throw % > 75% of mean:** As above ● **Add RON per 100 Minutes:** Approximated metric of offensive pressure ● **Add RON per Possessions:** As above, using possessions ● **Remove Points per Game:** Does not indicate whether player was a good player for other reasons e.g. assists, applying offensive pressure, etc ● **Remove 3P%, FP% and FT%**: Newly added metrics may be more meaningful
1.c. Experiment Objective	1. Research NBA domain knowledge to engineer features based on insights 2. Setup MLflow to track experiments and test combinations to conclude which feature engineering choices result in the best performance. Based on results, a final combination of options will be chosen to use in further experiments. By having an MLflow tracking environment, the aim is to be able to make more reasoned observations about experiment results. It will also be a practical review of how well it assists with an experiment and what issues arise with its use.

2. EXPERIMENT DETAILS

Elaborate on the approach taken for this experiment. List the different steps/techniques used and explain the rationale for choosing them.

2.a. Data Preparation

CSV was read into dataframe and copied into cleaned version to preserve original input. ‘TARGET_5Yrs’ column separated as target label. ‘Id_old’ and ‘Id’ columns were removed as they are not data features.

Many negative values noted: 5,216 values across 4,954 rows. These were certainly incorrect given our domain knowledge that Games Played, 3-Pointers-Attempted, etc cannot be less than zero. They were converted to `np.nan` before being imputed using sklearn’s Iterative Imputer. Iterative Imputer, which generates values based on neighbouring features, was used as it seemed detrimental to simply add the mean. Unlike more linear problems, the feature set of human athletes interplay in much more complex ways.

Dataset was standardised using StandardScaler as value ranges were quite disparate. Data was then split into training and test sets, then cross validated for scoring.

As listed in Section 1.b. Hypothesis, four data preparation options were made available to toggle, which are repeated here:

Impute negative values, Upsample with smote, Downsample randomly, Recalculate percentages

Instead of a scikit-learn Pipeline, a custom function + dictionary were used to set options as this method interopped with MLflow more seamlessly.

For implementation, see:
https://github.com/ronvoluted/kaggle-nba/blob/master/src/features/data_features.py

2.b. Feature Engineering

As listed in Section 1.b. Hypothesis, twelve feature engineering options were made available to toggle, which are repeated here:

Impute negative values, Upsample with smote, Downsample randomly, Replace % column data with own recalculations, Add Possessions, Add Points per Possessions, Add 3-pointers per 100 Possessions, Add Field Goals per 100 Possessions, Add Free Throws per Games, Add 3-Pointer % > 75% of mean, Add Field Goal % > 75% of mean, Add Free Throw % > 75% of mean, Add RON per 100 Minutes, Add RON per Possessions, Remove Points per Game, Remove 3P%, FP% and FT%

Much research was performed in the domain of basketball statistics and NBA analysis to garner these theorised choices. The main observation was that statistics based on time spent with the ball are more important than those based on time on court, number of games played, or even points per game.

Not all statistics tracked by NBA analysts were available in the dataset, so some approximations were necessary. For example, ‘possessions’ were calculated as:

rebounds + steals

Usage Rate was renamed Relative Offense Number (R.O.N.) and based on:

(field-goals-made + three-pointers-made) * free-throw-attempts * assists * turnovers

For implementation, see:
https://github.com/ronvoluted/kaggle-nba/blob/master/src/features/data_features.py

2.c. Modelling

A logistic regression model was used for all tests as it proved appropriate and reliable for this dataset in the previous experiment. Default hyperparameter values were used, as comparison of models was not part of the hypothesis. As such, this is a controlled variable in the experiment.

---

3. EXPERIMENT RESULTS

Analyse in detail the results achieved from this experiment from a technical and business perspective. Not only report performance metrics results but also any interpretation on model features, incorrect results, risks identified.

3.a. Technical Performance

(Note that MLflow UI only presents metrics to a precision of 3)

Results: Baseline

A baseline run was performed with all options set to false:

	AUROC	Accuracy
Test set	0.692	0.842
Training set	0.7	0.835

Results: Single options

16 runs were performed with only 1 of each of the options enabled. The purpose was to determine whether any options resulted in worse scores, but surprisingly no options gave an AUROC training result that was > 0.0007 below baseline. This is not a conclusive difference and did not even show up in MLflow UI (only in exported CSV due to rounding to 3 decimal places), so no options were considered ‘bad’.

AUROC training mean: 0.701683185

Notably, enabling downsampling alone resulted in the highest AUROC by far, though at the cost of accuracy (likely due to high false negative rate).

	AUROC	Accuracy
Test set	0.715	0.707
Training set	0.716	0.695

The next highest scoring result was much closer to baseline, and involved enabling upsampling alone.

	AUROC	Accuracy
Test set	0.702	0.84
Training set	0.703	0.833

Since the business objective is to score highly on AUROC the lower accuracy is acceptable, though this could still be an important insight for future experiments and would not be ideal for real world production use.

Results: All options with downsampling

A run with all options enabled was performed (besides upsampling, which would be redundant after downsampling).

	AUROC	Accuracy
Test set	0.672	0.7
Training set	0.706	0.696

As expected, simply combining every single option is not automatically the best approach. Interestingly, this did result in a higher-than-mean score however.

Results: Only greater than baseline options

A run was performed with the only options that resulted in greater than baseline AUROC training scores, as indicated in the “Single Options” runs. These were:

- Downsampling
- Adding player Possessions column
- Adding Field Goals Reliable column
- Removing Points per game column
- Removing percentage columns

	AUROC	Accuracy
Test set	0.67	0.678
Training set	0.705	0.69

Despite the logic in only using ‘positive’ options, the results are less than when simply downsampling.

Results: Only greater than baseline options + recalculating percentages

While recalculating percentages alone did not result in a greater than baseline score, as previously determined, the < 0.0007 difference does not necessarily preclude it as ‘bad’.

Since the dataset includes negative percentages which are certainly errors, fixing/recalculating these values should result in better results. This run uses the same options as the previous run, with the addition of recalculated percentages.

	AUROC	Accuracy
Test set	0.666	0.688
Training set	0.707	0.69

As hypothesised, there may be an improvement when imputing new percentages.

Results: Recalculating percentages and downsampling

Since it was established that enabling as many options as possible was not ideal, a run was tested with only 2 options.

	AUROC	Accuracy
Test set	0.666	0.683
Training set	0.714	0.701

Tese results seem to suggest that downsampling alone produces the highest score, and any other added option will only ‘dilute’ the score.

UPSAMPLING

Since upsampling was the 2nd highest scoring single option, for thoroughness the above downsampling runs were also tested with upsampling instead.

Results: All options with upsampling

	AUROC	Accuracy
Test set	0.684	0.838
Training set	0.702	0.83

Results: Only greater than baseline options

	AUROC	Accuracy
Test set	0.701	0.839
Training set	0.702	0.834

Results: Only greater than baseline options + recalculating percentages

	AUROC	Accuracy
Test set	0.693	0.842
Training set	0.703	0.834

Results: Recalculating percentages and upsampling

	AUROC	Accuracy
Test set	0.694	0.841
Training set	0.703	0.833

These combinations exhibited the same patterns as when downsampling was used, the only difference being lower scores on average.

Conclusion

Despite testing multiple theories of combinations, the highest scoring result was given by simply downsampling the dataset with no further engineering performed.

Approximating real world NBA metrics using incomplete statistics produced mixed-to-negative results.

Balancing the dataset resulted in a marked improvement, especially via downsampling. The reduction in samples did not appear to lead to heavy overfitting.

There were only two options that removed columns and enabling any combination of them always resulted in improved results. This suggests the provided features are not 100% useful/needed for the business objective, even with errors imputed.

3.b. Business Impact

Use of MLflow Tracking was immensely useful for the experiment and essentially enabled the approach of comparing feature engineering combinations. Continued use will improve the flow and organisation of experiments going forward.

For domain knowledge feature engineering that will make decent improvements in results, a more experienced NBA analyst/hobbyist will be required.

May need to submit a PR to MLflow to see more granular metrics in UI (see 3.c. Encountered Issues)

There is potential to improve the current best Kaggle submission, given previous local scores were lower and also not cross validated.

3.c. Encountered Issues	Basketball statistics is a huge domain and often requires an experienced analyst to produce meaningful insights. The attempts in this experiment could be seen as amateur, and not all statistics used in modern NBA analysis were available in this dataset. MLflow UI is hardcoded to only display numbers to a precision of 3. The project's maintainers have indicated this is by design and not shown interest in changing it since mid-2020. This is problematic as AUROC scores such as 0.700000 and 0.700444 presented identically as 0.700. For important measurements such as baseline and mean, the exact metric could be retrieved from an exported CSV, but this was unfeasible for frequent run comparisons. Scikit-learn Pipelines were considered for setting the options as they seem like a natural fit. However, the syntax for Pipelines and their method for setting options did not work smoothly with MLflow's **`log_param`** and **`log_metric`** methods.

4. FUTURE EXPERIMENT	
Reflect on the experiment and highlight the key information/insights you gained from it that are valuable for the overall project objectives from a technical and business perspective.	
4.a. Key Learning	a) MLflow is very effortless to use yet extremely useful in what it sets up b) Testing combinations of feature engineering options can still be very manual c) NBA feature engineering based on domain knowledge is difficult and it may not be possible to gain deep strides in performance with this approach d) Downsampling and upsampling the dataset resulted in the highest and second highest scores respectively. This dataset and business objective are very suited to balancing
4.b. Suggestions / Recommendations	MLflow should be used in almost every data science project from now on. Explore if combination testing could be 'automated' in a similar fashion to how hyperopt generates the best combination of hyperparameters. After downsampling, optimise the logistic regressor's hyperparameters with hyperopt. This will combine the learnings from experiment B and C to hopefully reach the best result achievable. Possessions could include scoring as part of the calculation and not just rebounds + steals. To do so, scoring could be approximated with: **scores** = 3-pointers made + field-goals made + free-throws made **possessions** = scores + rebounds + steals