Capstone Two Report
AUDL Ultimate Frisbee: Score Margin and Win Probability Prediction

Background

The American Ultimate Disc League (AUDL) was established 2013 in North America, and currently consists of 24 men's teams across four regional divisions competing professionally. The AUDL modified some standard rules of Ultimate Frisbee to increase marketability. Most importantly, games are not played until a certain score is reached, but are defined by timed quarters. If both teams have the same score at the end of regulation, then they play up to two overtime periods to declare the winner. Sport specific discussion will be limited in the main body of the report, but the reader can refer to Appendix I – Statistics Glossary for a description of game flow and context of game summary statistics.

Problem Statement

Eleven years of game results, summary statistics, and player data are now available with the conclusion of the 2023 season. This project seeks to use the most basic game summary statistics to predict the outcome of the game: the winning team and the difference of the two teams' scores. The final models should provide a basis for understanding the other summary statistics, and the data pipeline should exemplify collecting, cleaning, and exploring AUDL data for future studies.

A league executive may manage the AUDL's investment into data collection and publishing. The final model should add value to existing data, and could be used to predict games for the upcoming season and establish odds It can also check past statistics for validity. Bad data limits the potential of the model, therefore the executive may find it worthwhile to standardize bookkeeping across games and to backfill historical games without valid statistics. The AUDL leads other Ultimate Frisbee organizations in data and footage collection, and is relatively open to sharing its data compared to other professional sports leagues. This allows for community engagement and projects that drive interest in the league – see AUDL Throw Explorer, unsure of author to attribute.

Another stakeholder may be a team coach or manager. The prediction model should provide quantifiable value to each game event, and thus helpful in used developing and optimizing strategy. For example, individual players or their combinations, "lines", could be graded for each point they play. The stakeholder can use the numerical assessments of the model alongside their visual assessments of the game to determine playing time distribution and situational line calls.

AUDL Data

Collection

Game statistics were collected from the AUDL website using their REST API (documentation). For this project, one endpoint was used with date specification to retrieve a list of all games played and their game IDs (URL). Each game ID was used with another endpoint to collect game statistics (example URL). The JSON returns were read into pandas DataFrames, and cleaned data was persisted in the Parquet format. Each record is a unique game, specified by its game ID, with six descriptive features, and thirty numerical features. Records for 1,604 games were collected. Refer to Appendix I – Statistics Glossary for detailed feature information.

Cleaning

Data was thoroughly cleaned, explored, and checked prior to model building [1]. Descriptive features (date, location, team names, etc. . .) were not used as final model inputs, but were helpful in evaluating results. Some of these features were used to demonstrate one-hot-encoding and dummy feature creation. For example, **week** was encoded and could be used as a season chronology feature.

Missing values were assessed, and each feature was checked to ensure that values of **0** made sense and were not null placeholders. For example, if either **throws** or **completions** were not recorded (missing or 0), then the other was also marked as suspicious. Following a similar process of feature-specific checks and explorations, other games were found with faulty records. Duplicate games were not present, but the data checks yielded some dubious records. It is very unlikely that both **home** and **away** teams identical game summary statistics. Such records likely indicate that one of the two team's stats were not recorded. Refer Appendix III – Data Check Details for more on this process.

1,604 game records on import → missing features → 1,547 games → faulty records → 1,521 games

Automated outlier detection was explored on the cleaned dataset using Local Outlier Factor and IsolationForest algorithms provided by scikit-learn. Rudimentary visualizations of the outlier detection algorithms indicated possible utility as a final data cleaning step – data checks may not have been exhaustive. Scaling data prior to detection seemed to improve Local Outlier Factor, but not IsolationForest (graphs in repository).

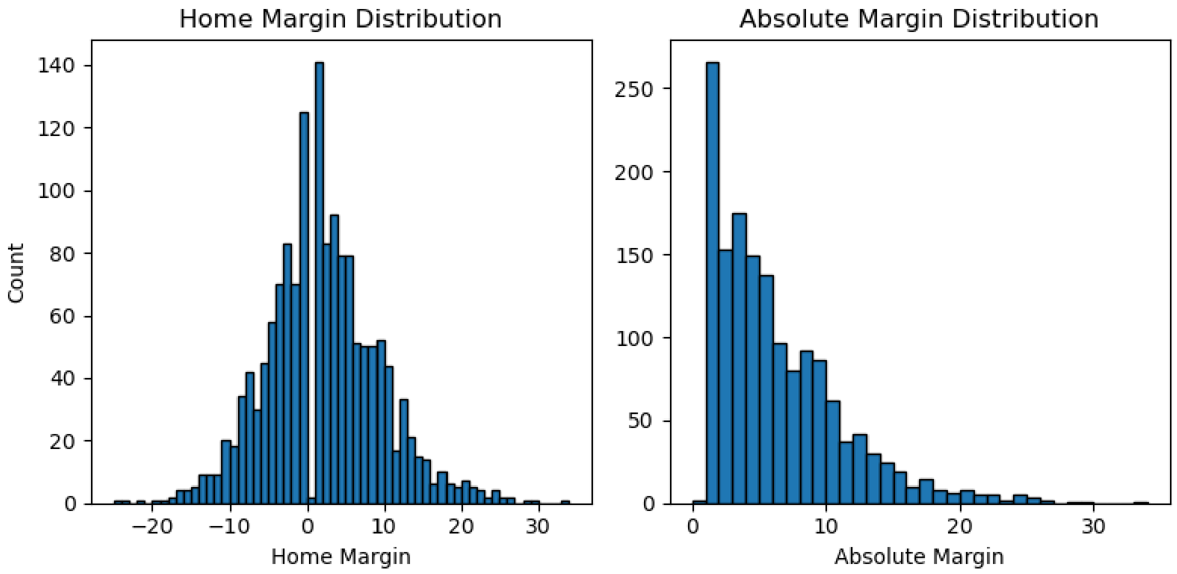
Selection

Before removing records with missing or faulty data, "basic" features were identified. Because the scoring team will almost always start the next point on defense, the final score of the game can be analytically determined from a number of features (Appendix I – Statistics Glossary). Therefore only **throws, completions, blocks,** and **turnovers** were kept as basic features and used for model inputs**.** The other features will not be presented in the report, but can be seen in the EDA notebook. Culling non-basic features left a much greater portion of records with complete data, especially as some game statistics were not introduced until 2019 [2]. The original data size is already limited, and further reduction through cleaning is not ideal. Learning curve analyses will be presented in the Final Results section, and should provide motivation for improved bookkeeping.

Target Features

Target features were defined using **home score** and **away score**. The continuous target for regression models, **home margin**, is the difference of the two. The binary target for classification models, **home win**, is true if **home score** is greater than **away score**. While **home** and **away** teams provide a convenient basis for building a team-independent dataset, each target showed a slight "home-field-advantage" bias. It will be assumed that the home team's advantage is not inherent, and that the other game features will capture its effect.

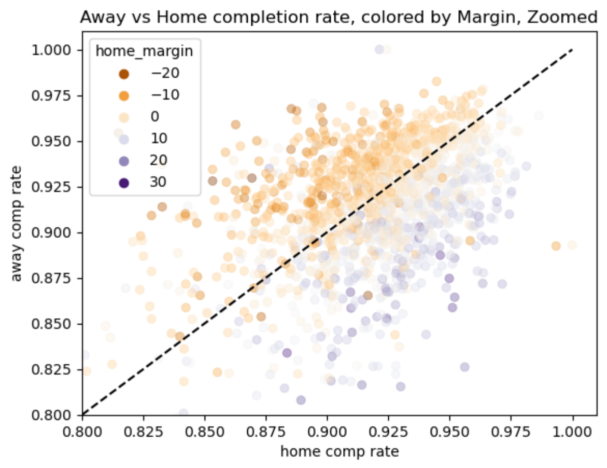
Statistical tests implied a real difference between overall home and away score distributions, but provided less clear results when home-field advantage was assessed for each team.

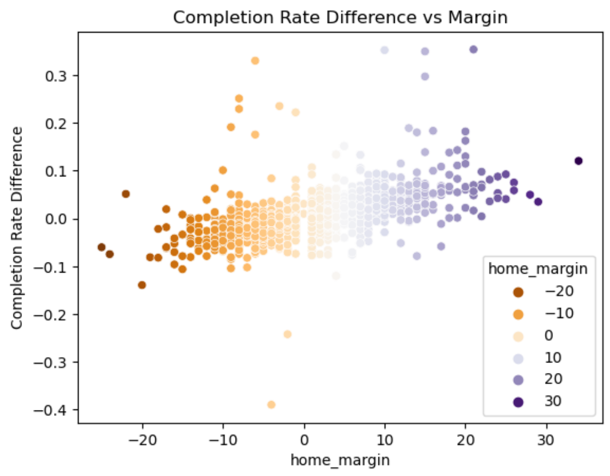


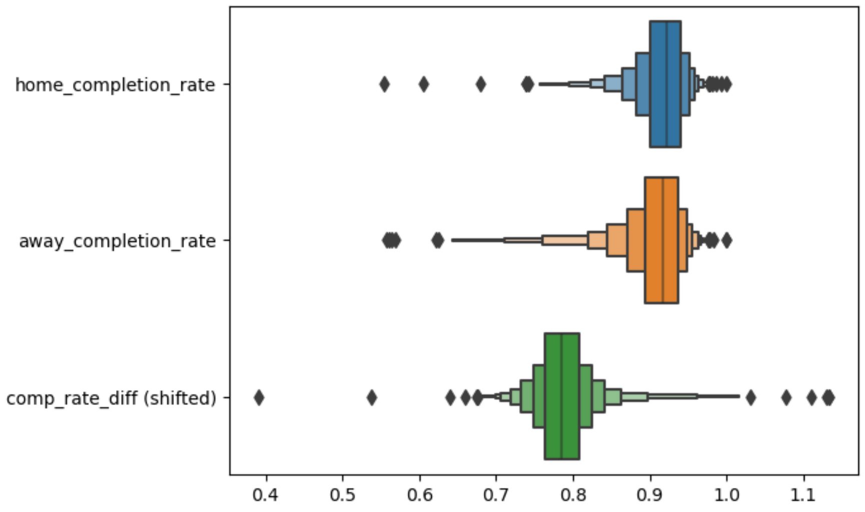
Home margin is not quite symmetrical, as shown on the left histogram, and favors the home team. Accordingly, the home team won more often than the away team (58%). This discrepancy may make sense as teams are less likely to bring their full rosters to away games, and is worth keeping note of when assessing eventual prediction errors. Ties are rare (2 total), but were not specially accounted for in prediction models.

Feature Engineering

Basic features were combined in an effort to better correlate with targets and to maintain useful interpretability. **Completion rate** was calculated for each team, and also the **completion rate difference** between the two teams. Features describing a change in possession were aggregated, **blocks** minus **turnovers**, for a given team; and the **block-turnover difference** between competing teams was included. The following **completion rate** graphs show the logic and motivation behind creating these features (see also: **block-turnover** graphs). **Completion rate difference** relates more strongly to the target features and is more normally distributed than its components features.

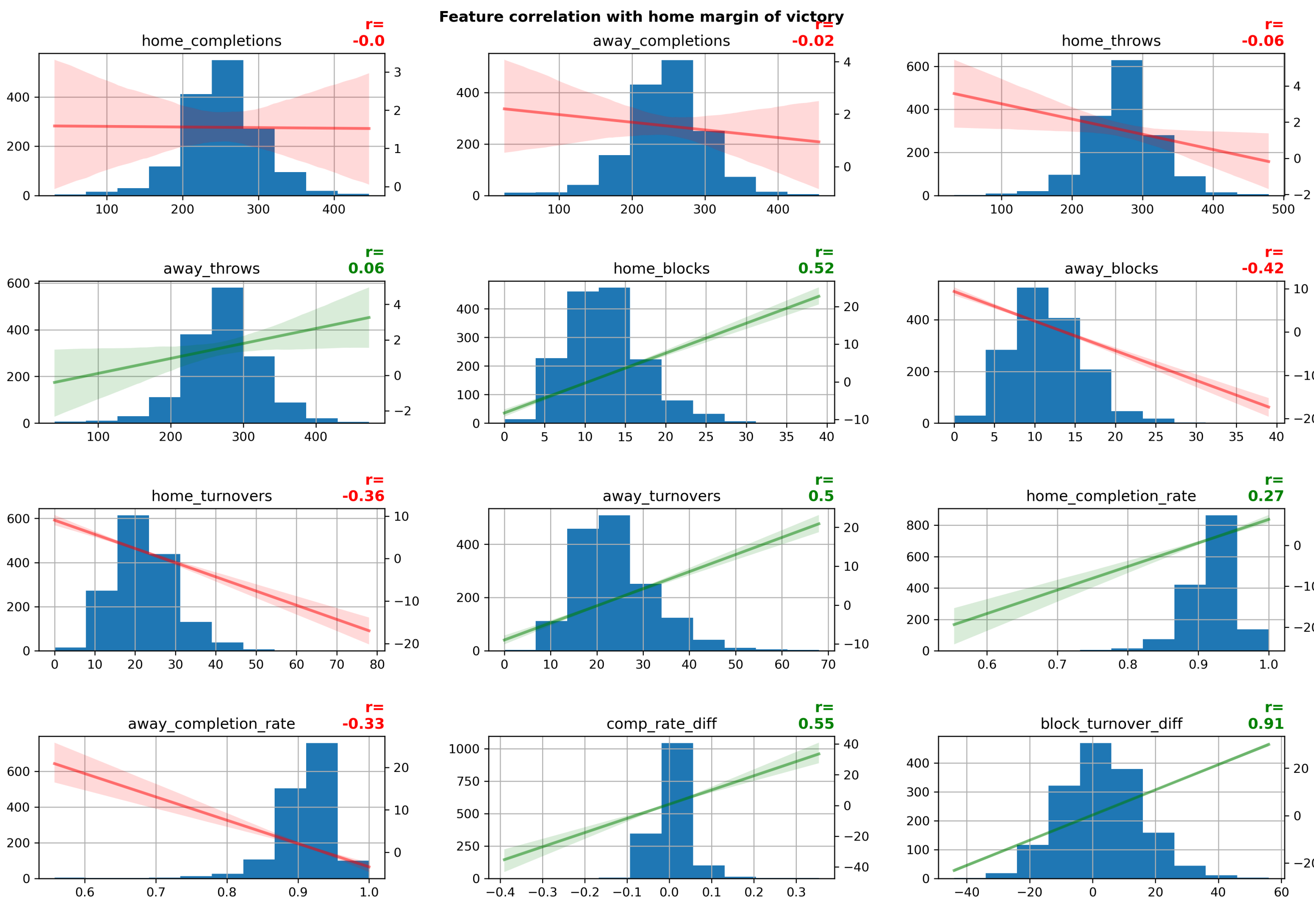




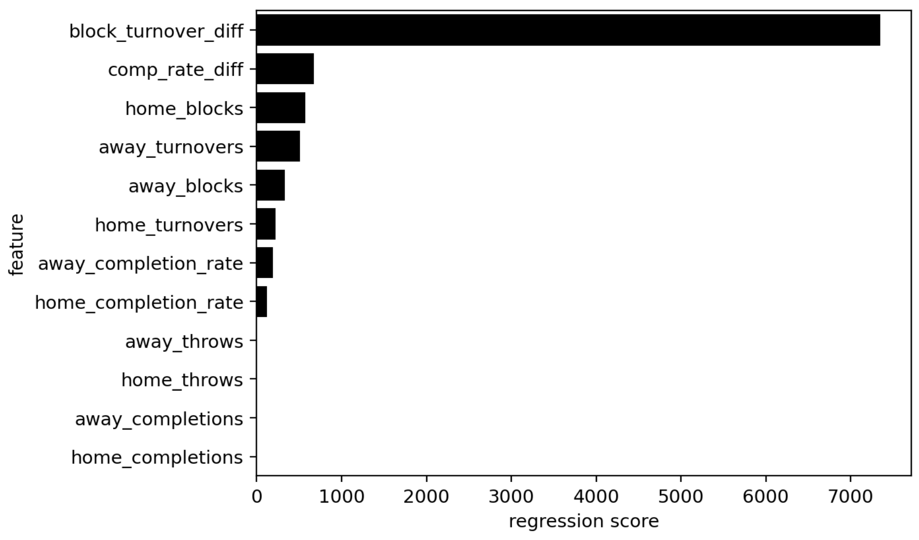


Exploration

Feature distributions before and after cleaning steps were examined, and feature correlation with each target variable was assessed [3]. The histograms below show that most features are distributed somewhat normally. The **home margin** regression lines and correlation coefficients indicate the engineered features relate more strongly to the target than their components. Intercorrelations between the features were assessed with a correlation heatmap [4]. The strongest correlations stem from inherent relationships, such as **throws** and **completions**. Collinearity was addressed during pre-processing for certain models, and feature interactions were assessed with the final model [12].



x-axis: feature values *left y-axis: histogram bin count* *right y-axis: home margin target*



F-statistic and p-values were used to score features for regression (**home margin**) and classification (**home win**) [5]. The scores highlight the importance of features involving possession change and show that **completion rate difference** brings throw and completion features into relevance with **blocks** and **turnovers**.

The final 1,521 records, now with twelve features, were split 4:1 into training (1,216) and testing (305) sets for model evaluation. A random seed was set to ensure consistent splits and repeatable model evaluation across studies. Target feature stratification was not imposed with the split, but was confirmed after the fact [6]. Data was not normalized prior to training, as the effect of normalization was studied for each model.

Model Training

A number of machine learning algorithms were evaluated, with optimal pre-processing conditions and tuning parameters determined for each one. This report will focus on the regression models predicting **home margin**. Classification models were evaluated with a similar process (notebook). The following metrics were used to score the regression models in the studies presented in the report.

Model Metrics

Name	Abbreviation
Coefficient of determination	R2
Mean-squared error	MSE
Root mean-squared error	RMSE
Root mean-squared log error	RMSLE
Mean absolute error	MAE
Mean absolute percentage error	MAPE

Initial studies investigated feature selection and z-score data normalization. Grid search cross-validation was used to evaluate linear regression models: linear, ridge, lasso, stochastic-gradient descent (SGD), elastic-net, least-angle, orthogonal matching pursuit, kernel ridge, Bayesian ridge, automatic-relevance-determination, Huber, passive-aggressive, random sample consensus (RANSAC); and a k nearest neighbors model (kNN). The first parameter grid spanned the range of features (1-12), which were selected in order of their regression scores above.

The average MAE, RMSE, and R2 cross-validation test scores were used to measure performance. Scores were typically correlated, allowing simple comparison, but MAE rankings were not always consistent with the other metrics. Linear models may minimize their MAE with more features, but do not generalize as well. Some of the outlier-resistant models (Huber, RANSAC) had consistently lower MAE than other models, as might be expected, and had more unique trends with performance vs features. Generally 5-10 features showed the best overall performance for the better models [7a]. After determining the best number of features for each model, they were evaluated with raw and normalized data. The following models outperformed the others and were progressed for thorough* hyperparameter tuning: kNN, kernel ridge, ridge, and SGD regression. Each was best with seven or eight features; kernel ridge and kNN performed better without normalization.

Boosting and bagging ensemble models were studied separately than the models above, as they were expected to perform better using all features. The regression models included GradientBoosting, AdaBoost, Random Forest, and Extra Trees from scikit-learn, and also CatBoost, LightGBM, XGBoost (linear boosting), XGBoost, (tree boosting). After confirming that these models were best with all features, the effect of normalizing data was studied. Average cross-validation test scores showed that normalization typically improved or did not affect performance for the better models [7b]. CatBoost, LightGBM, Extra Trees, GradientBoosting, Random Forest, and XGBoost (both) scored well and were progressed for more thorough* hyperparameter tuning and direct comparison with linear and kNN models.

**Hyperparameter tuning was briefly explored using PyCaret default grids in the above studies.*

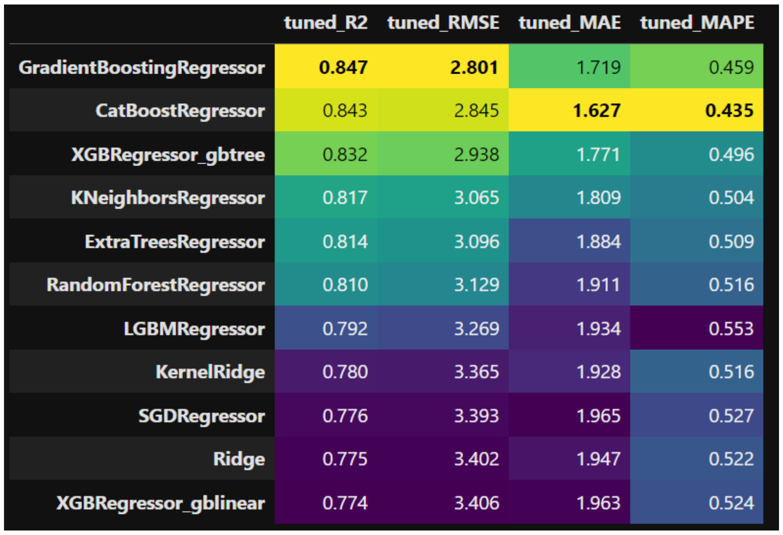
Models progressed for tuning are summarized in the table below, along with their specific pre-processing conditions and source packages. Test scores following tuning are also included on the table, and will be discussed in the following section. Note that the numbers in the table are relative to each other: **1** is the best of the bunch, and the others' distance from one describes how much worse they scored than the best. Actual values are presented in the Hyperparameter Tuning section.

Models for Tuning Study + Relative Results

Name [name in graph]	Pipeline	Package	Tuned Test Metrics (relative)*			
			R2	**RMSE**	**MAE**	**MAPE**
Ridge	Select 8 features, normalize		0.91	1.21	1.2	1.2
Kernel Ridge	Select 7 features	scikit-learn	0.92	1.2	1.18	1.19
StochasticGradientDescent (SGD)	Select 7 features, normalize		0.92	1.21	1.21	1.21
kNN [KNeighbors]	Select 7 features		0.96	1.09	1.11	1.16
RandomForest			0.96	1.12	1.17	1.19
ExtraTrees		scikit-learn	0.96	1.11	1.16	1.17
GradientBoosting			**1**	**1**	1.06	1.06
LightGBM [LGBM]	All features, Normalize	LightGBM	0.93	1.17	1.19	1.27
CatBoost		CatBoost	0.99	1.02	**1**	**1**
XGB (tree) [XGB_gbtree]			0.98	1.05	1.09	1.14
XGB (linear) [XGB_gblinear]		XGBoost	0.91	1	1	0.95

Scores in the table are relative to each other: [model score ÷ **best score] | Final Models combined into VotingRegressor*

Hyperparameter Tuning



	tuned_R2	tuned_RMSE	tuned_MAE	tuned_MAPE
GradientBoostingRegressor	**0.847**	**2.801**	1.719	0.459
CatBoostRegressor	0.843	2.845	**1.627**	**0.435**
XGBRegressor_gbtree	0.832	2.938	1.771	0.496
KNeighborsRegressor	0.817	3.065	1.809	0.504
ExtraTreesRegressor	0.814	3.096	1.884	0.509
RandomForestRegressor	0.810	3.129	1.911	0.516
LGBMRegressor	0.792	3.269	1.934	0.553
KernelRidge	0.780	3.365	1.928	0.516
SGDRegressor	0.776	3.393	1.965	0.527
Ridge	0.775	3.402	1.947	0.522
XGBRegressor_gblinear	0.774	3.406	1.963	0.524

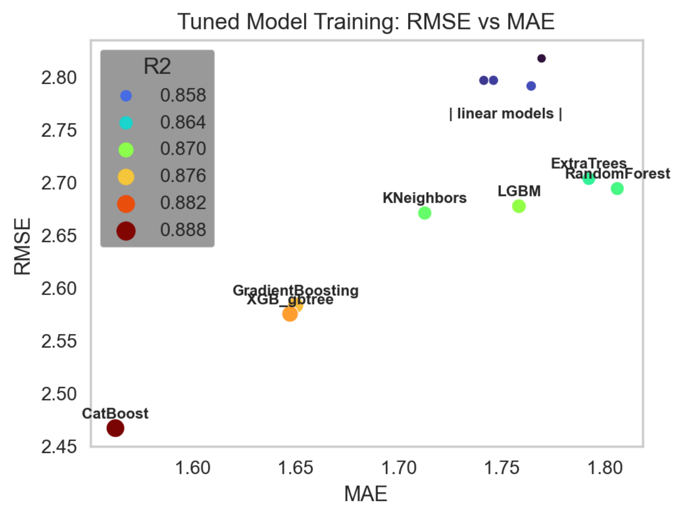
Hyperparameters were tuned using **RandomizedSearchCV** from scikit-learn. For each model, parameter grids (helpful tool) and iteration counts were determined such that some combination of R2, MAE, and RMSE improved in a reasonable amount of time. Tuned models were then evaluated by cross-validation (training) and test set scores. Metrics for models were mostly correlated, although MAPE was not always suitable for testing, as division by zero can occur with the **home margin** target.

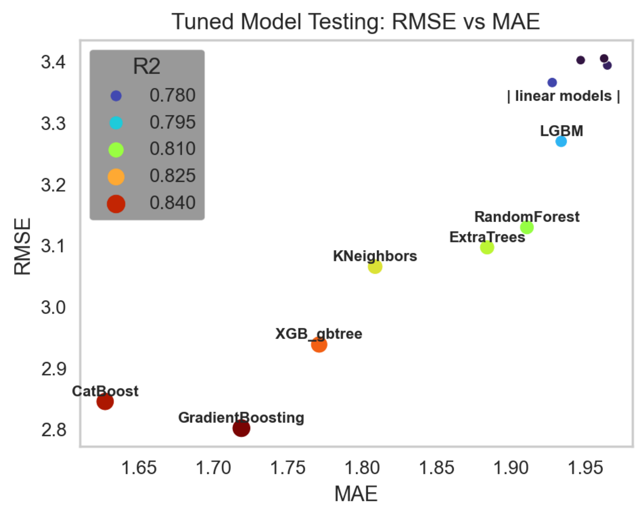
tuned models scored on the test set **Best | Okay | Bad | Worst**

Training and testing scores were compared to evaluate overfitting. Most models showed a similar trend for the two scores but LightGBM did not generalize as well as the others [8]. Parameter grids, tuning results, and scoring tables are located in the repository.

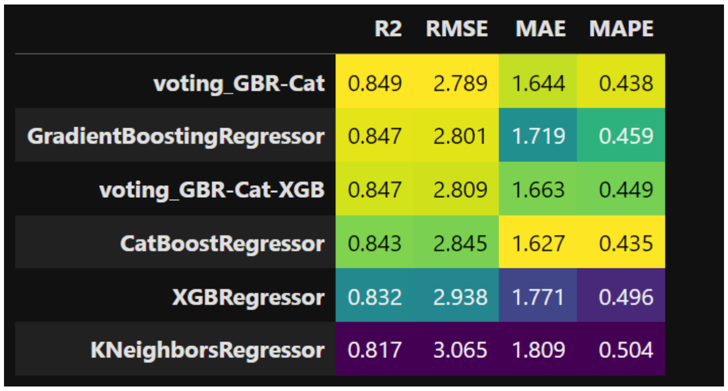
The graphs below show training, left, and testing, right, R2 (marker color), MAE (x-axis), and RMSE (y-axis) for the models. The linear models performed poorly compared to ensemble models and kNN. CatBoost, GradientBoosting, and XGB (tree based boosting) showed the best performance. The test scores on the right graph are also shown in the table above.

better models are ↙ | testing results (right) used for final evaluation (and shown in tables)





Lastly, blends of the top models, VotingRegressor meta-estimators, were created to see if performance could be improved further. Tabular results of the best models and selected blends are shown below.

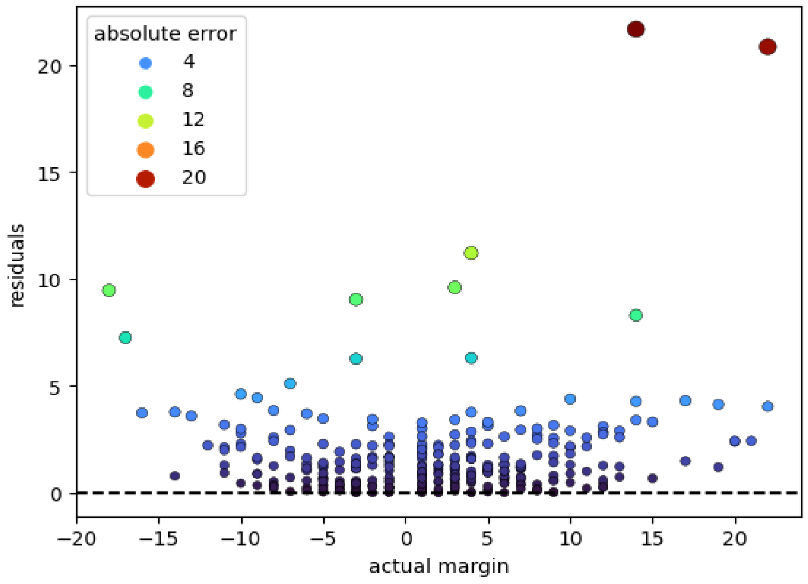


	R2	RMSE	MAE	MAPE
voting_GBR-Cat	0.849	2.789	1.644	0.438
GradientBoostingRegressor	0.847	2.801	1.719	0.459
voting_GBR-Cat-XGB	0.847	2.809	1.663	0.449
CatBoostRegressor	0.843	2.845	1.627	0.435
XGBRegressor	0.832	2.938	1.771	0.496
KNeighborsRegressor	0.817	3.065	1.809	0.504

The CatBoost+GradientBoosting blend provided the best result, it improved some aspects of each of its component models. In a sesnse, it harmonizes the better generalization of GradientBoosting and the lower error of CatBoost. In practice, the residual analysis was almost identical for the final model blend and the CatBoost model alone. See the final model submission file for all model parameters and metrics.

Final Results

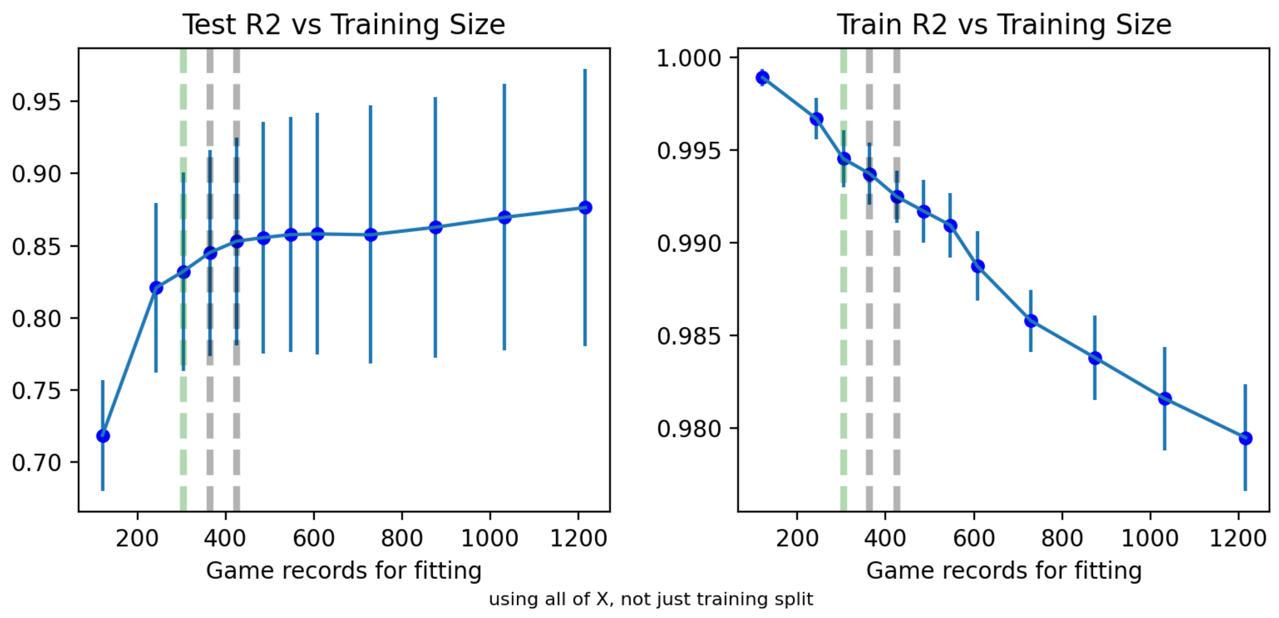
Game summary statistics provided from the AUDL's API were used to create reasonably accurate models to predict the team's difference of scores and the winning team. Predictions could be improved, but performance was limited by design. Only the most fundamental statistics, and their meaningful combinations, were used for model building, and now their importance to winning can be interpreted from the model [11]. As mentioned, these values can be used to measure the impact of players or individual game events. The league can use the model to create a win probability indicator or predict future game results.



Additionally, the model's worst predictions are indicative of invalid records that should have been removed during cleaning [9] [10]. Some of the game pages confirm missing records, but the API may still return bogus data. Given the utility of the game summary statistics, these games could be flagged for review by the league and corrected at leisure with the game footage. More importantly, it can ensure that teams are recording statistics consistently for future games and publish classification guidelines.

The prediction error seems to be normally distributed, but the positive **home margin** bias may effect games with the lowest actual margin. This trend is more apparent when looking at predicted margin vs actual margin [9].

Learning curve *analyses for the final model: First dashed line (20%) shows the actual data used for training. The next two lines (25%, 30%) show potential for improvement. Similar trends are apparent for [MAE](MAE) and RMSE.*



Additionally, model performance is limited by data size. A 4:1 split was used to provide enough records for a fair testing set, but learning curve analyses indicate that more training data could have improved model performance. Recall that five percent of game records dropped due to invalid data. Continued data collection for the model will improve performance, as would retroactive data cleaning.

Lastly, a repeatable data collection, cleaning, and evaluation process were established. Obvious indicators of invalid data have been identified and faulty game records can be documented to streamline future work.

Future Recommendations

The final model should be expanded to include team and match-up specific predictions. Team labels were ignored for this analysis, but the model's errors were not team independent [13]. New models could target the input features for the current model, such that the outcome of future games, say the upcoming 2024 season, could be predicted. Additionally, data pipeline can be adapted for online learning as the season progresses.

The majority of features returned in the game summary statistics were not used for the win margin and win probability models. While these features were not desirable as aggregate game statistics, increasing their granularity to a per-point or per-possession basis will allow deeper exploration. Throws and completions were found to have little to no value for this study, but considering their finer distributions, and what results those lead to, may increase their predictive power.

Much more data is available from the AUDL API, such as roster information for each point and positional tracking (XY) of the disc and each of its events. This work provides a template to collect such data and assess more problem statements.

https://www.theaudl.com/stats/glossary

Appendix I – Statistics Glossary

Introduction

Games are played between two teams. A team is on offense when it has possession of the disc. The offense can move the disc around the field with throws. An incomplete throw, or holding the disc for too long, results in a change of a possession. If a team catches the disc in the opposing endzone (regardless of which team threw the disc), the point ends with that team scoring. The scoring team then starts on defense and "pulls" the disc to other team, which will start on offense. This is similar to a kickoff in American Football, or like football, if the starting team booted the ball as far as they could to the other team. Games are considered similarly to tennis: the team starting on **offense** is expected to score (hold), and teams often consider their scores on **defense** (breaks) as importantly as the overall **score**.

The final score for each game was used to create the **Target Features**, and some of the **Standard Features** were combined to create **Engineered Features.** Features were renamed for clarity and brevity during import and again for this report. See the Data Wrangling notebook for original data returns and feature names. **Descriptive Features** that appear in the figures and discussion are also mentioned.

Standard Features

The following are provided with each game summary. Statistics are provided twice, once for the **home** team, and once for the **away** team. Some terms have the additional distinction of **offense** and **defense** (O/D), indicating that a given team *started* the point on O/D, then earned the statistic. A coin toss determines the whether the **home** or **away** team starts on **offense** or **defense**. After the first point (with the exception of each quarter start), the scoring team will start on defense and the team that didn't score will start on offense.

Term	Description	Insight
Score*	Final score for a given team.	The home and away score together describe the final result. Used for target feature definition.
Throw	Pass attempt. Player throws disc.	The team with the disc changes their position by throwing to one another.
Completion	Successful pass attempt, thrown disc is caught by the same team.	Most throws in the AUDL are successful. Missed throws are significant.
Blocks	The team without the disc gains possession, *forced error(?)*	Unsure about distinction/overlap between blocks and turnovers. Data exploration raised more questions than answers.
Turnover	The team with the disc loses possession, *unforced error(?)*	
Hucks*	Long pass attempt, player throws disc	Unsure about cutoff. Default

	more than *(?)* yards.	distance metric for the AUDL is yards. Hucks were not tracked until 2021.
Hucks Completed*	Successful long pass attempt by the same team.	
(O/D) Score*	Successful point for a team. One team scores in a point, one does not.	Scoring on offense is considered a “hold”, and scoring on defense is considered a “break”.
(O/D) Point*	Simply a count of points played for a team, starting on O/D.	
(O/D) Possession*	Count of possessions each time a team is playing a point. Scoring without a turnover is one possession.	Unlike scores and points, possessions could be considered a “basic” statistic and incorporated in the models. However, it is more valuable in combination with features that were removed.
Redzone Possessions*	Possessions occurring within a certain distance *(?)* of the endzone (scoring target).	Unsure about cutoff. Redzone statistics were not tracked until 2021.
Redzone Scores*	Successful redzone possessions.	

** feature not used as model input*

Target Features

Target features were designed to be independent of team and to avoid double-counting of games. Each game was considered from the home team’s perspective.

Term	Description	Insight
Home Score Margin	Home score minus away score. Continuous target for regression models.	Ties, resulting in a margin of 0, are rare by design of overtime periods. This was not explicitly input to the continuous target.
Home Win (Probability)	Home win is true if the home score is greater than the away score. Binary target for classification models.	Instead of considering ties as a result, the binary classification is maintained by *not* considering ties as a win.

Engineered Features

Some standard features were combined to emphasize their importance on game outcomes.

Term	Description	Insight
Completion Rate	A given team’s completions divided by their throws. Normalizes throws and completions across games.	Imbalance between implied missed throws (blocks, turnovers) and reported throws suggests inconsistent classification.
Completion Rate Difference	Difference between two team’s completion rates. Home team minus away team (home team’s perspective).	Distribution much less skewed than throws, completions, or completion rate.

*(H/A) Blocks + (A/H) Turnovers**	Overall turnovers for a team. The home team's blocks are added to the away team's turnovers, and vice-versa.	Blocks and turnovers may not be consistently classified.
Block-Turnover Difference	Overall turnover difference for the two teams. Home team's perspective (positive correlation with target features).	Downstream effects of variable classification

** intermediate feature, not used as model input*

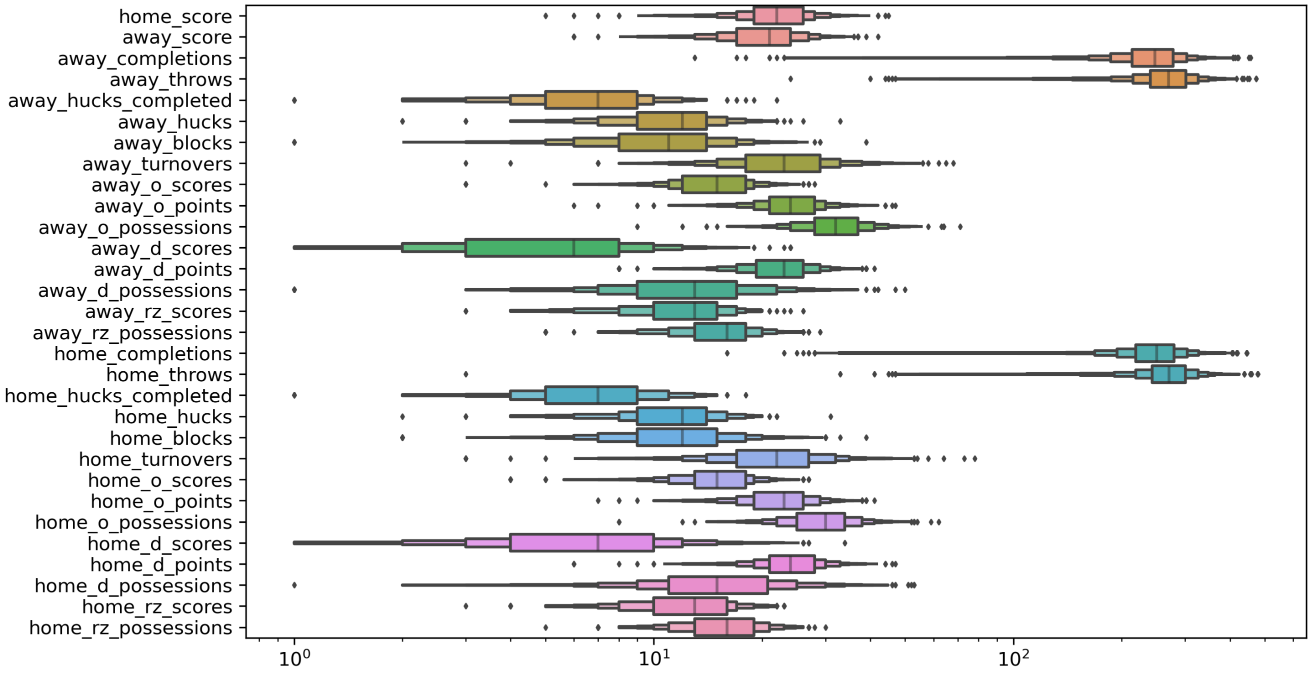
Descriptive Features

A few of the remaining features provided during import are shown below. For future analyses, some of these could be encoded and used for model inputs. For this study, they were simply used to group and analyze results.

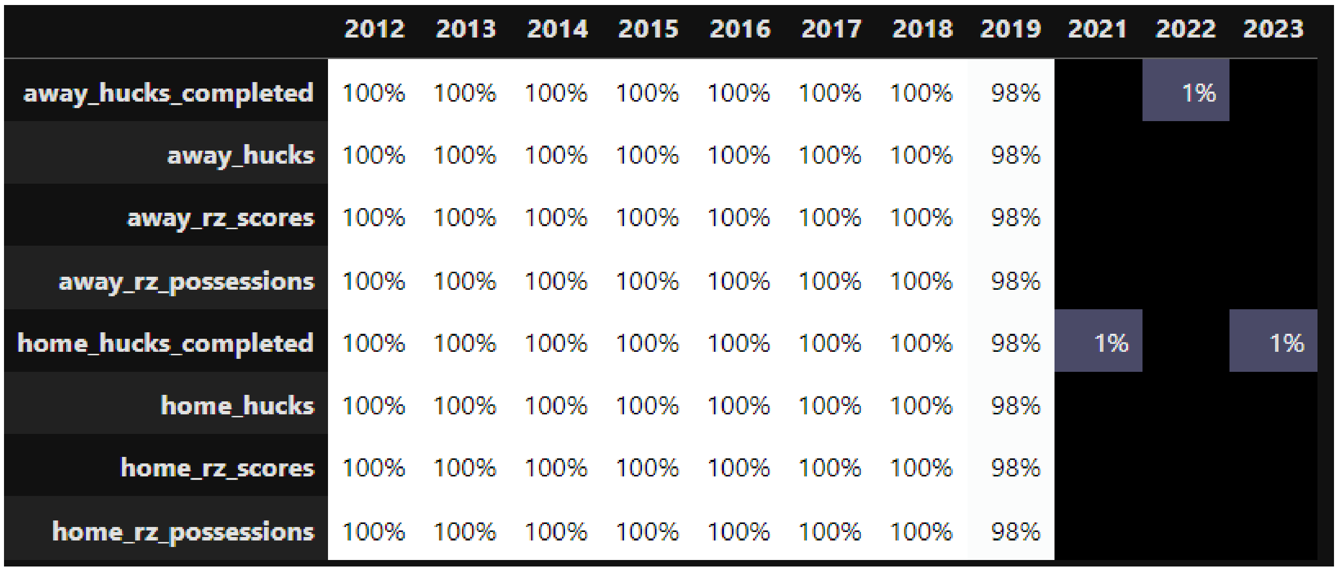
Term	Description
Date, Timezone	Timestamp for the game record. Start and end times provided.
Location	Location ID of the game, specific to actual venue and not team.
Week	Week of the regular season or play-offs. Indicator of season progression.
Home, Away	Teams playing in the game. See [13] for all teams.

Appendix II – Additional Figures

[1] Imported Feature Distributions – Boxplots show feature distributions prior to any data cleaning.

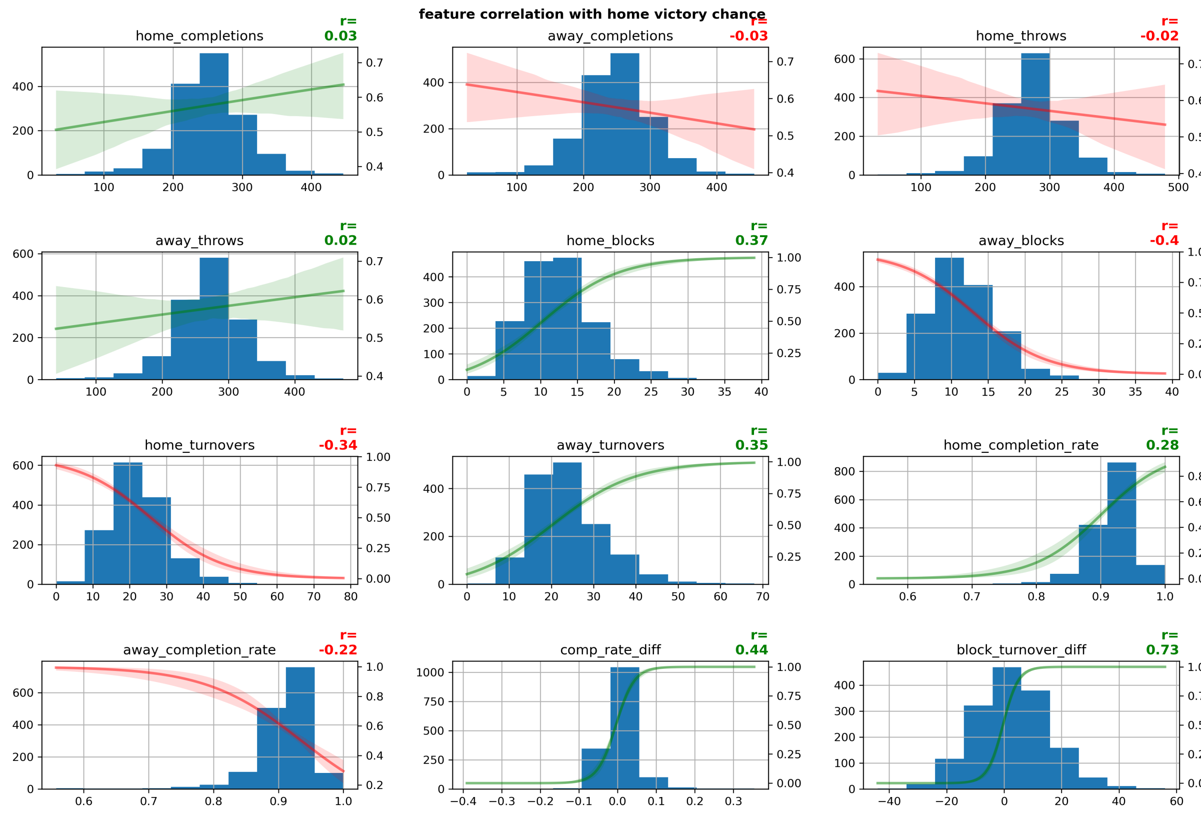


[2] Data Missing by Season – Hucks and Redzone statistics were not recorded prior to the 2019 season.

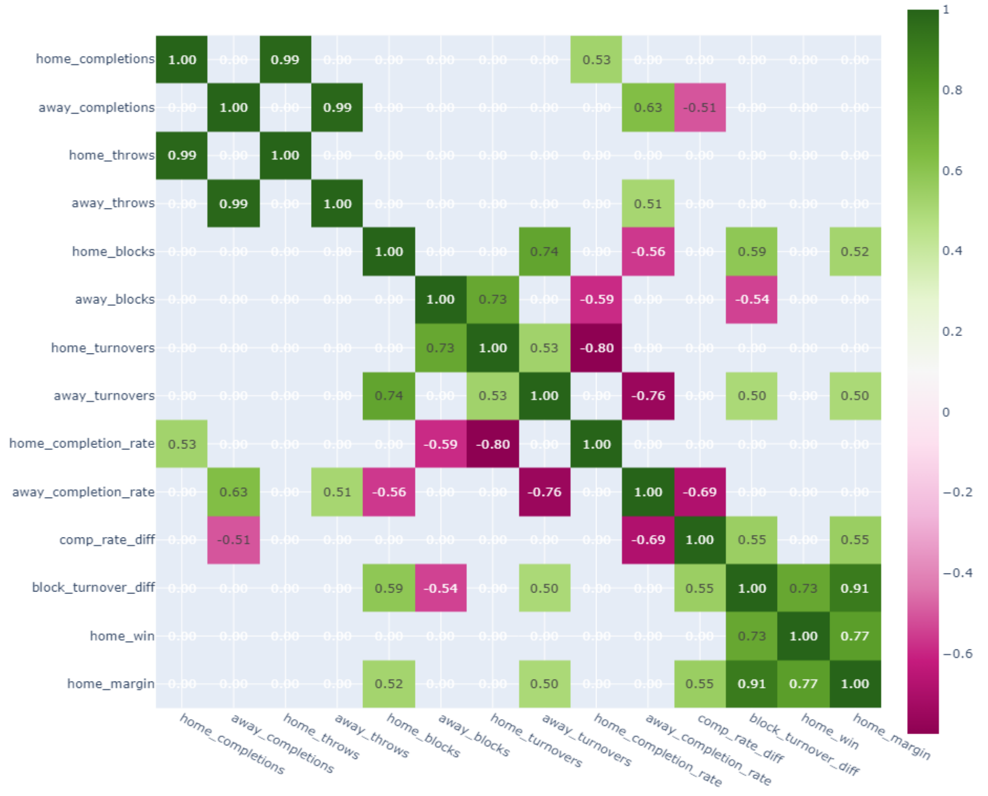


	2012	2013	2014	2015	2016	2017	2018	2019	2021	2022	2023
away_hucks_completed	100%	100%	100%	100%	100%	100%	100%	98%		1%	
away_hucks	100%	100%	100%	100%	100%	100%	100%	98%			
away_rz_scores	100%	100%	100%	100%	100%	100%	100%	98%			
away_rz_possessions	100%	100%	100%	100%	100%	100%	100%	98%			
home_hucks_completed	100%	100%	100%	100%	100%	100%	100%	98%	1%		1%
home_hucks	100%	100%	100%	100%	100%	100%	100%	98%			
home_rz_scores	100%	100%	100%	100%	100%	100%	100%	98%			
home_rz_possessions	100%	100%	100%	100%	100%	100%	100%	98%			

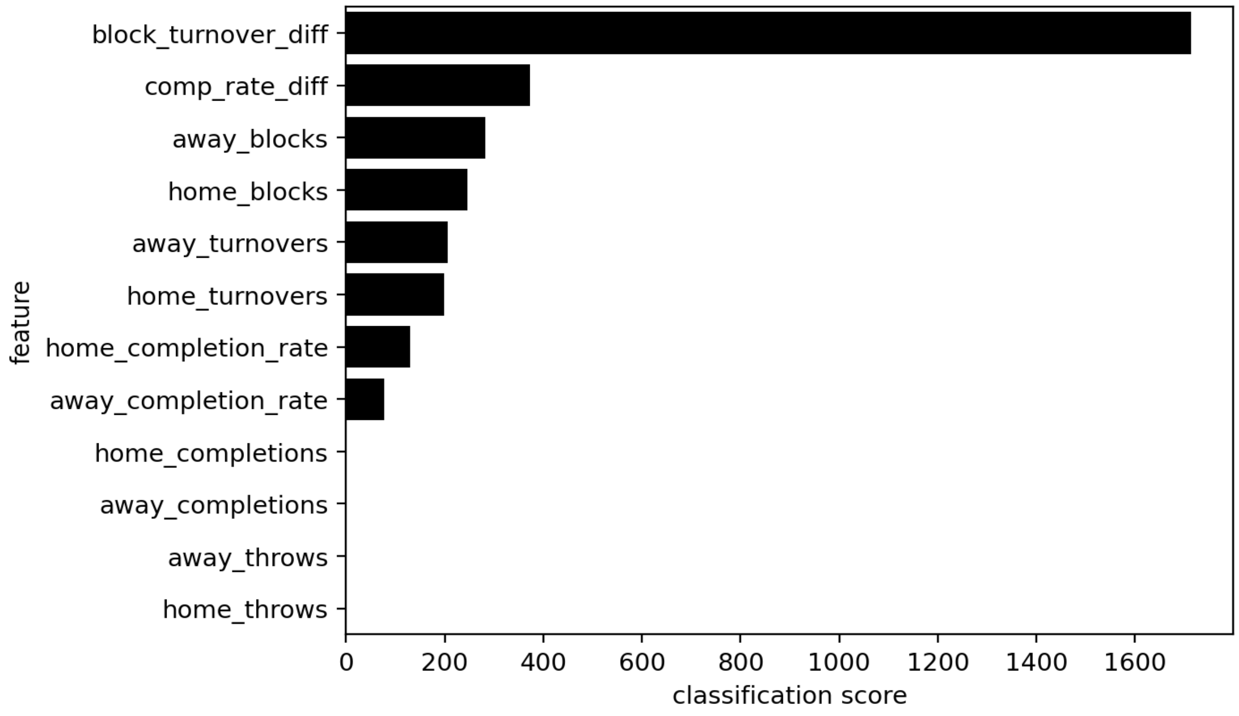
[3] Cleaned Feature Distributions, relation to Home Win – Histograms overlaid with logistic regression for **home win**.



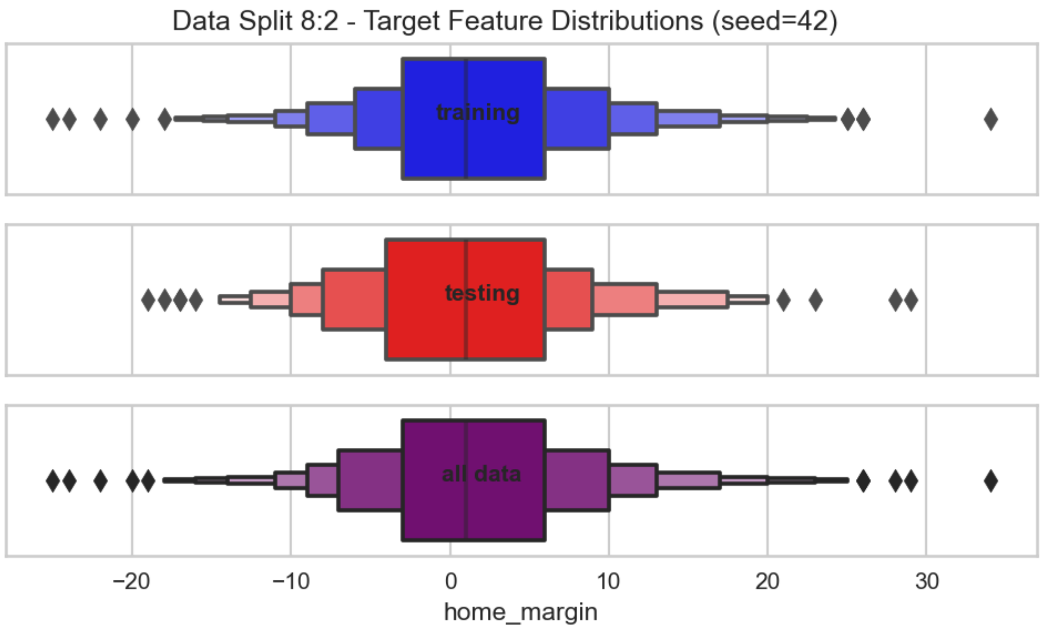
[4] Feature Correlation Heatmap – Color coded correlation coefficients for values > |0.5|



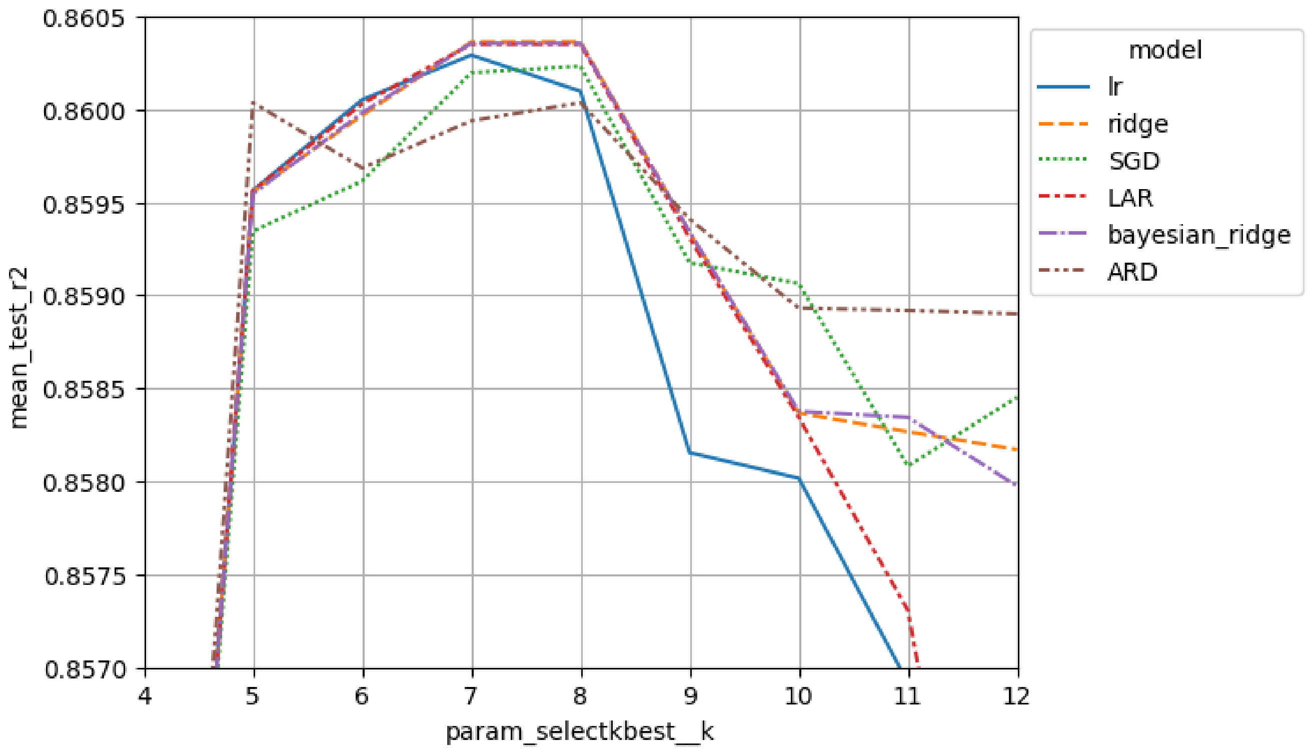
[5] Feature Significance for Home Win – Results from sklearn's SelectKBest with *f_classif* score function. Graph in report used *f_regression*.

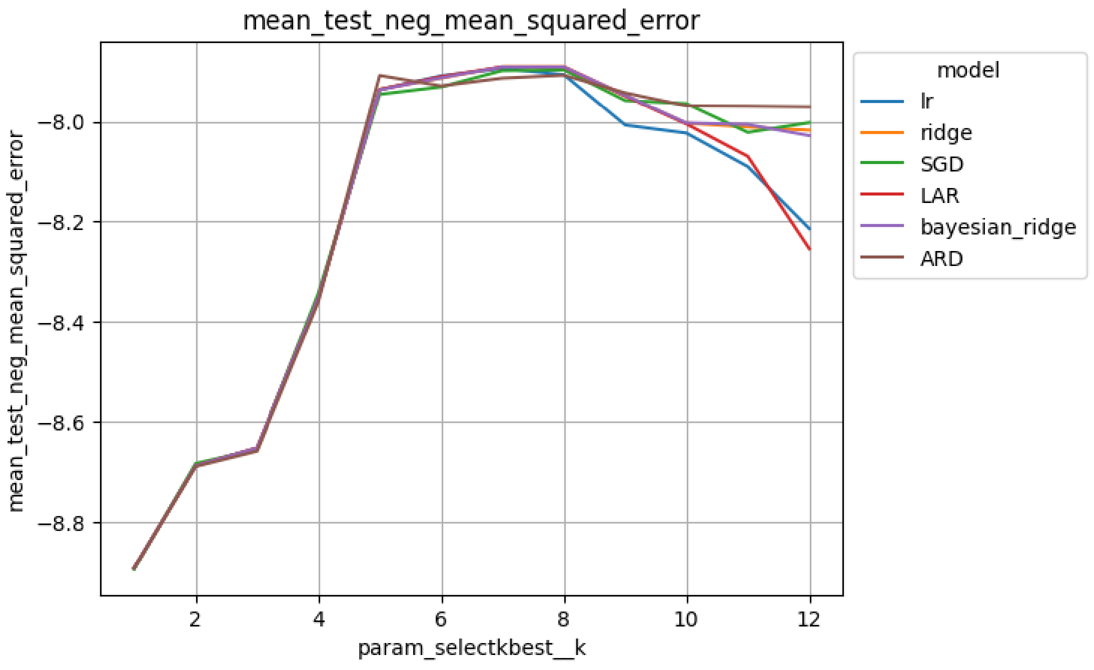


[6] Stratification Check – Target feature distribution in train and test sets.

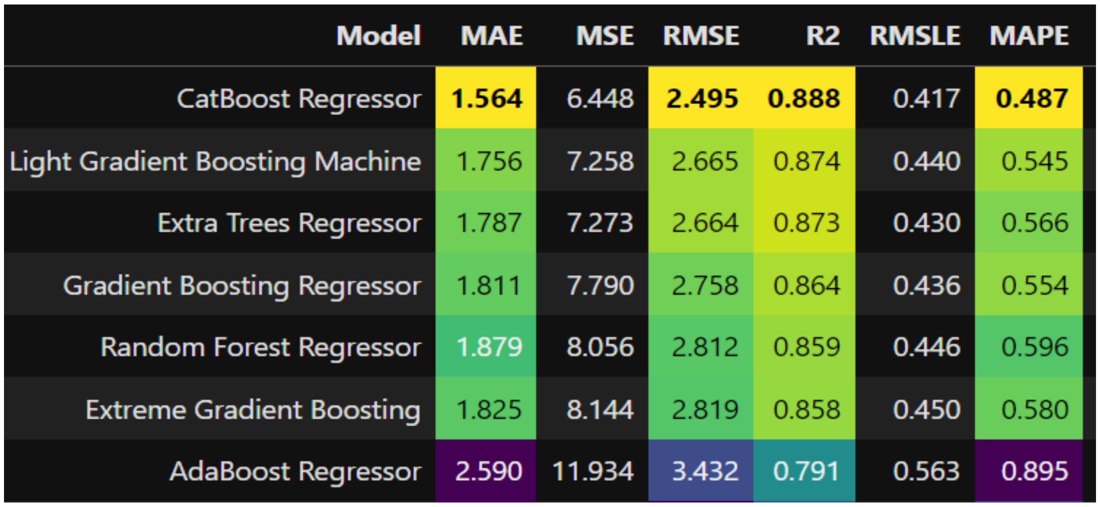


[7a] Linear Model Feature Selection – Model's R2 and MSE vs number of features selected. Top, R2, plot zoomed in on better models for emphasis. Other linear models show similar behavior. Outlier-robust and passive-aggressive models have unique trends with features selected.



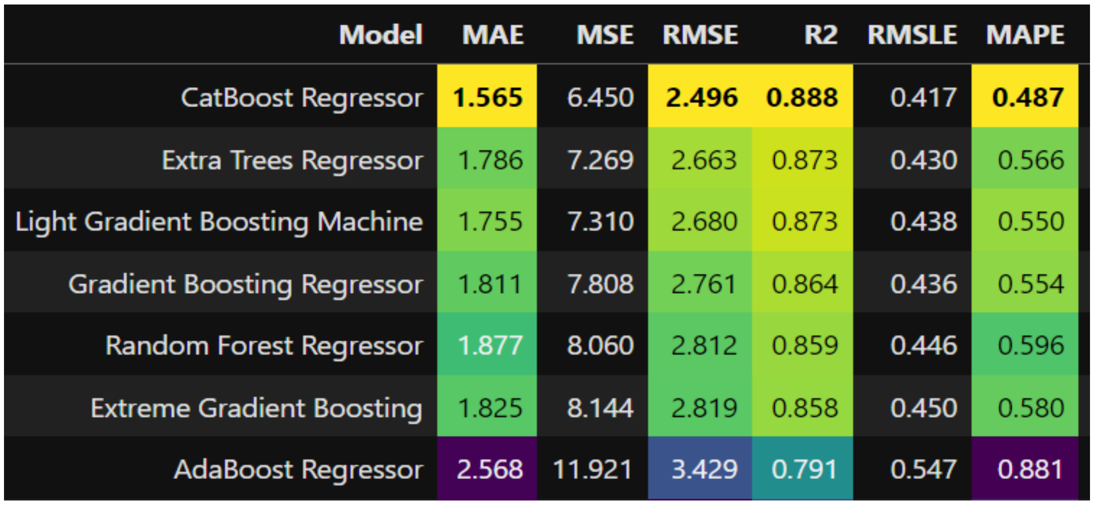


[7b] Ensemble Model Normalization Study – Cross-validation scores using normalized data (top table) and raw data (bottom table). Initial tuning with PyCaret used entire dataset. Manual hyperparameter tuning was more successful.



Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
CatBoost Regressor	**1.564**	6.448	**2.495**	**0.888**	0.417	**0.487**
Light Gradient Boosting Machine	1.756	7.258	2.665	0.874	0.440	0.545
Extra Trees Regressor	1.787	7.273	2.664	0.873	0.430	0.566
Gradient Boosting Regressor	1.811	7.790	2.758	0.864	0.436	0.554
Random Forest Regressor	1.879	8.056	2.812	0.859	0.446	0.596
Extreme Gradient Boosting	1.825	8.144	2.819	0.858	0.450	0.580
AdaBoost Regressor	2.590	11.934	3.432	0.791	0.563	0.895

↑normalized data↑ *↓raw data↓*



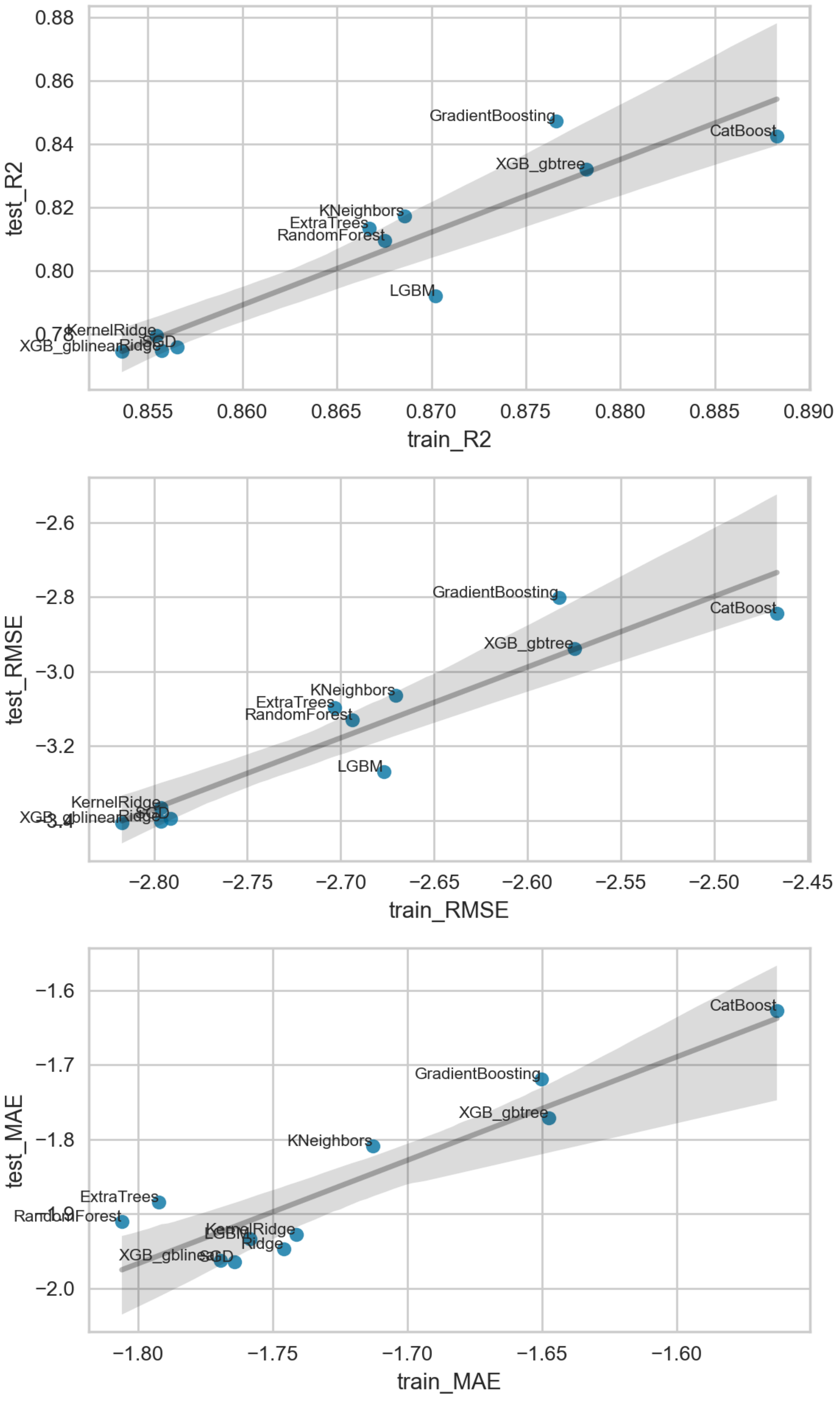
Model	MAE	MSE	RMSE	R2	RMSLE	MAPE
CatBoost Regressor	**1.565**	6.450	**2.496**	**0.888**	0.417	**0.487**
Extra Trees Regressor	1.786	7.269	2.663	0.873	0.430	0.566
Light Gradient Boosting Machine	1.755	7.310	2.680	0.873	0.438	0.550
Gradient Boosting Regressor	1.811	7.808	2.761	0.864	0.436	0.554
Random Forest Regressor	1.877	8.060	2.812	0.859	0.446	0.596
Extreme Gradient Boosting	1.825	8.144	2.819	0.858	0.450	0.580
AdaBoost Regressor	2.568	11.921	3.429	0.791	0.547	0.881

Entire dataset used for tables above, thus metrics are typically better than the test scores presented in report.

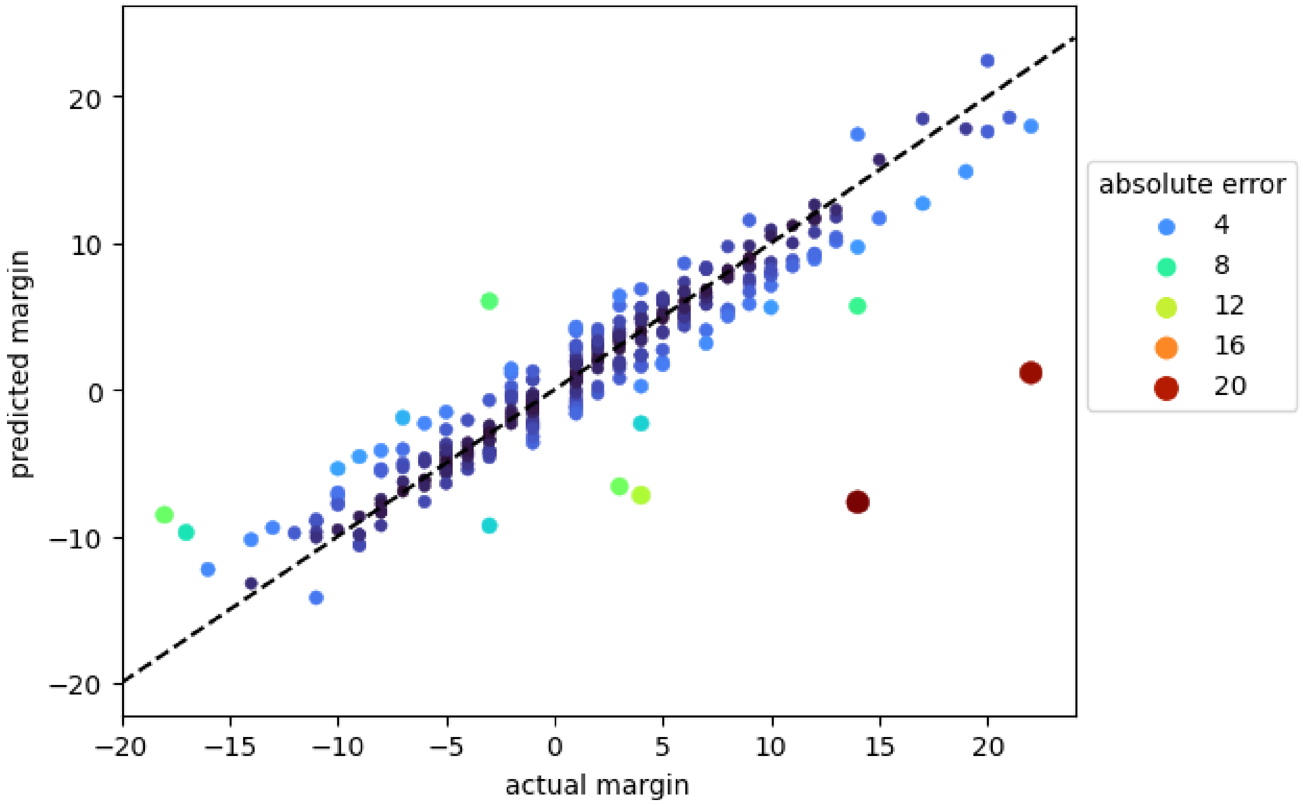
	CatBoost	Extra Trees	LightGBM	GBR	R.F.	XGB	AdaBoost
R2	+	-	+	+	+		-
RMSE	+		+	+	+		-
MAPE	+		+	+	-		-
MAE	+	-	-		-		-

Improved with Tuning | **Did not change** | **Worse with tuning**

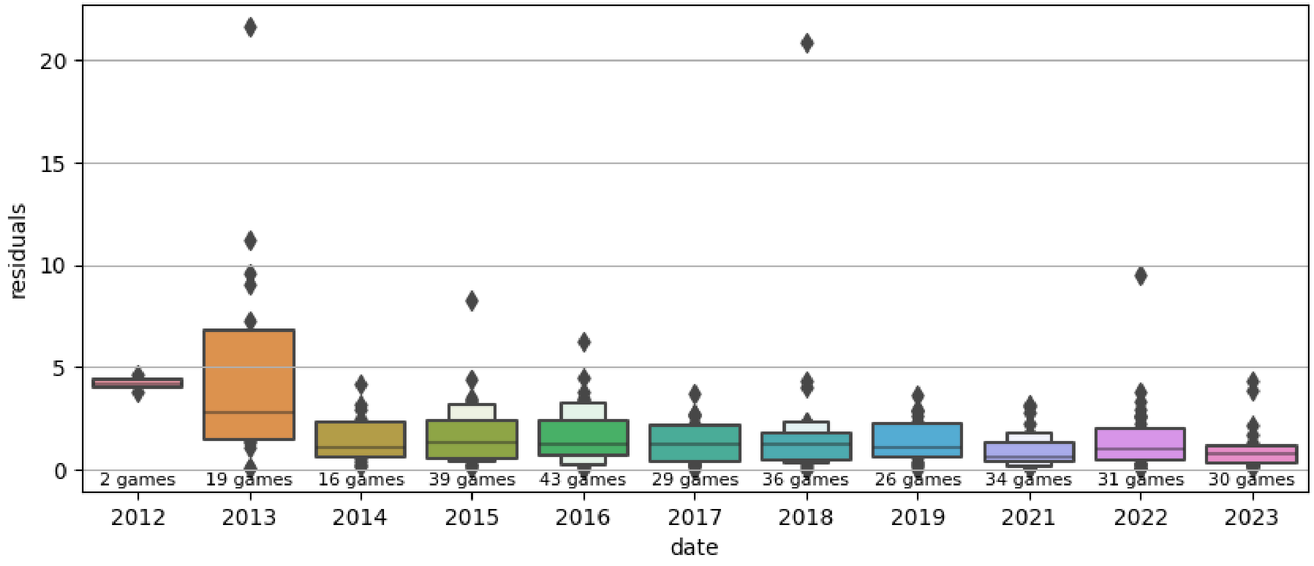
[8] Overfitting Analysis – Model test vs train scores for select metrics. ↗ indicates better performance



[9] Residual Analysis – Predicted margin vs actual margin. Points colored and sized by absolute error.

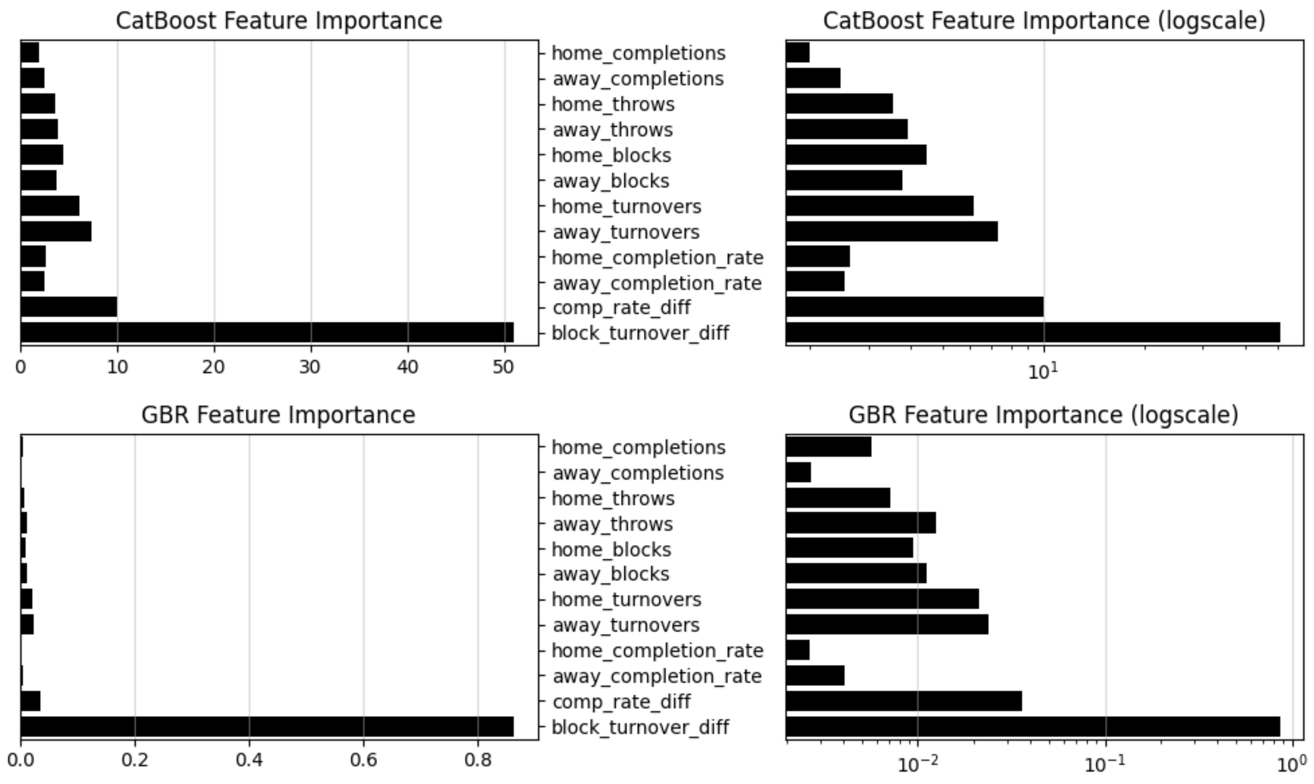


[10] Residual Analysis by Season – Residual vs Season for test data.

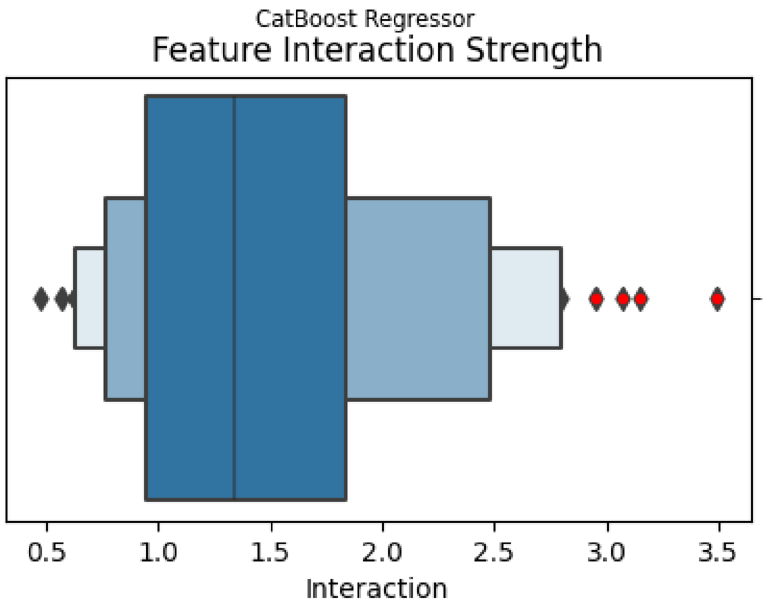


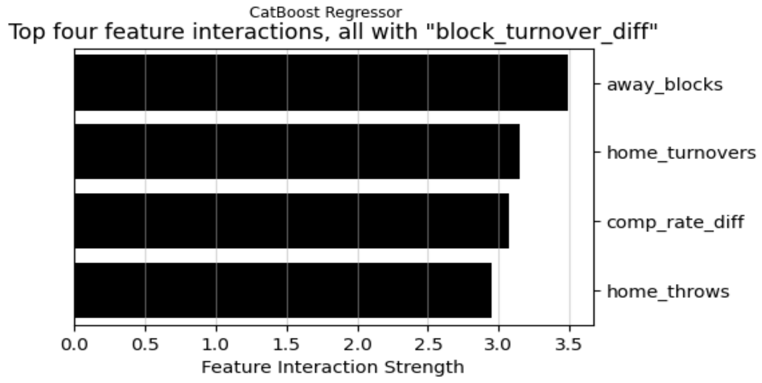
2013-05-04-DC-NY: predicted -8, actual 14, residual 22. Away_turnovers were unlikely to be 0, given home_blocks were 19.
2018-07-14-DET-PIT: predicted 1, actual 22, residual 21. Away stats were probably not recorded, only throws differ (208 vs 209) from home. This difference allowed record to survive cleaning.

[11] Top Models' Feature Importance – Feature importance for the two models comprising the final deliverable.

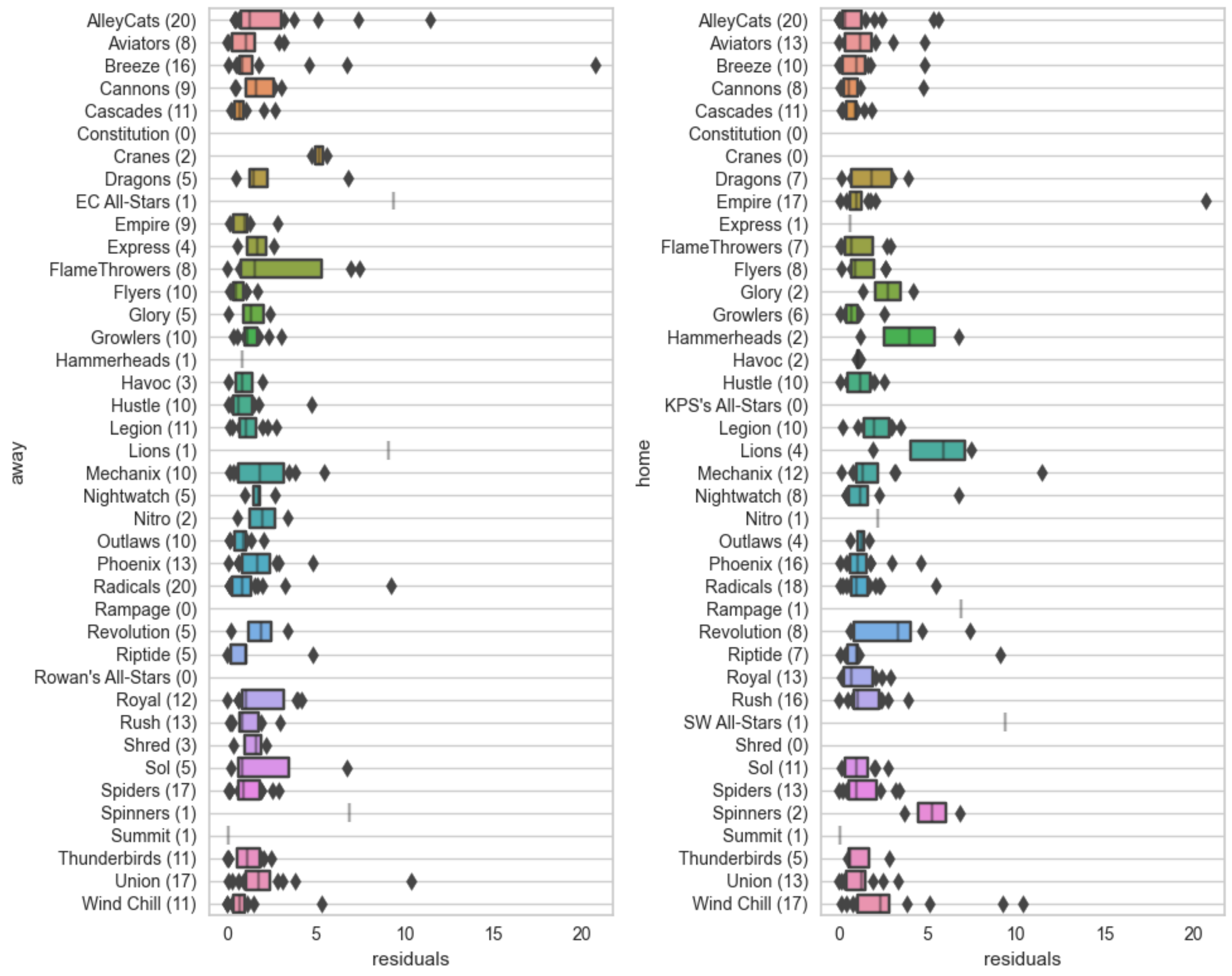


[12] CatBoost Feature Interactions – Feature interaction strength for the CatBoost Regressor. Top four feature interactions ◆ are presented in the bar chart below.





[13] Residual Analysis by Team – Test set error distributions for each team's home and away games.



Team name (game count)

Appendix III – Data Check Details

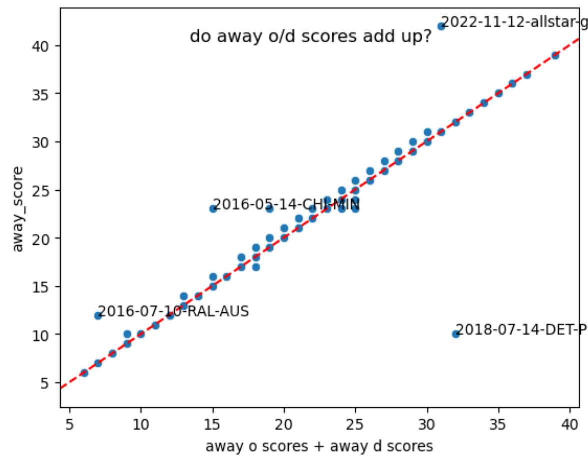
EDA Notebook

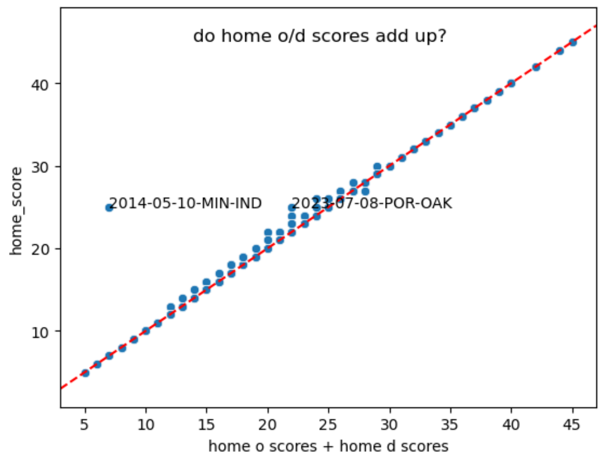
Missing Data (summary table on last page)
Initial data checks with slices of data removed: 2012 and 2013 seasons, huck and redzone statistics. Six games remained with incomplete data.

- 2014-05-18-SLC-SEA, 2014-07-27-SJ-TOR, 2015-06-27-CHI-DET, 2015-07-25-CHI-PIT, 2016-08-06-SEA-MAD, 2016-08-07-SEA-DAL

Next checks performed after only keeping the basic features. See final tables for summary and missing data by season.

Teams' O/D scores should add up to their total score *(check before removing features)*



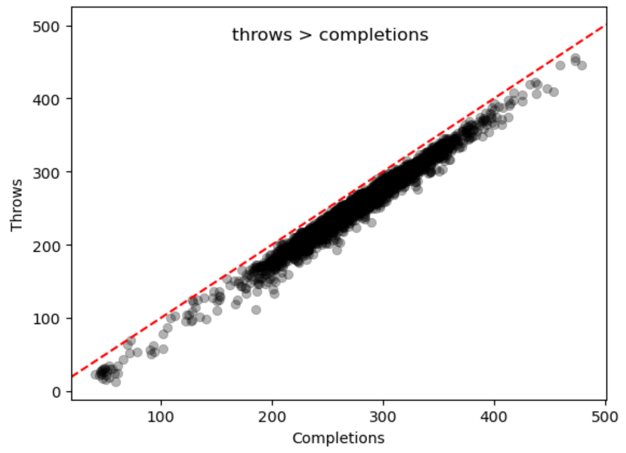


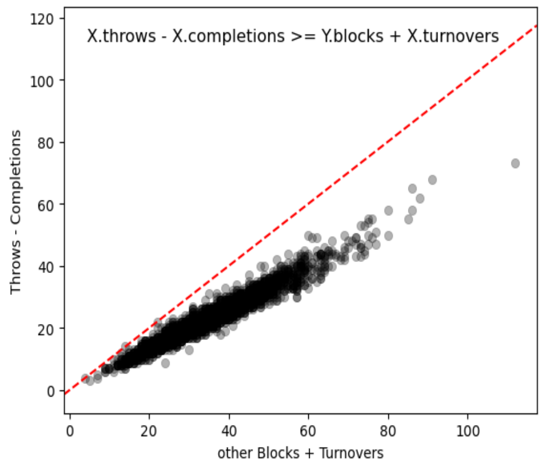
Points, possessions, score must add up

- Team's O/D points should be similar to opponent's D/O points
- O possessions ≥ O points, O/D points ≥ O/D scores
  - *D possessions vs points should be a valuable metric*

Throws, Completions, Turnovers, Blocks

- Throws should be ≥ than completions
- A team's throws minus completions should be ≥ a team's turnovers plus their opponent's blocks





Other checks, Outliers: **Outliers graphically identified by boxplots, then complete game records were inspected**

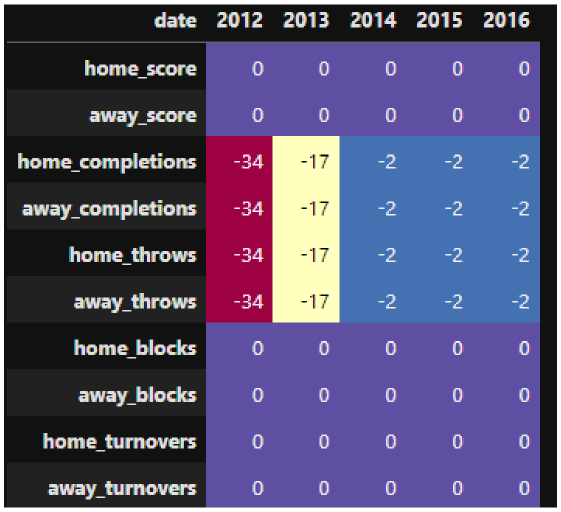
- Scores per completion << 1. Median value less than 0.2, but a few records show value close to 1.
  - home games for Cascades (team) may have only recorded throws that scored???
  - check games with > 0.9

- Turnovers per throw << 1. Median value less than 0.1, some outliers at 0.6-0.8
- Imbalanced stats for home and away: team's throws / opponent's throws > 6
- Some turnover outliers remained after previous cleaning
  - 2016-04-02-DET-CIN: other features indicate a game with truly bad offense or good defense
  - 2014-04-13-VAN-SLC: away_turnovers seem unlikely with low number of throws, to be dropped.

Record removal for final dataset (basic features only)

Step	Notes	**Games**
Initial Data Collection		**1,604**
Incomplete records	*Features missing or determined missing*	**-57**
Stats don't add up	*Turnover balance, incompletions*	**-3**
Suspect home bookkeeping	*Home completions = home score*	**-7**
Probably incomplete stats 1	*Home or away throws >> other*	**-7**
Probably incomplete stats 2	*Scores / completions > 0.9*	**-8**
Probably incomplete stats 3	*Turnover outlier (home)*	**-1**
Cleaned Data		**1,521**
Training Set	**1,216**	
Testing Set	**305**	

Incomplete records identified and removed in the first step above (57).
Most games without data were from the first two seasons, 2012-2013.

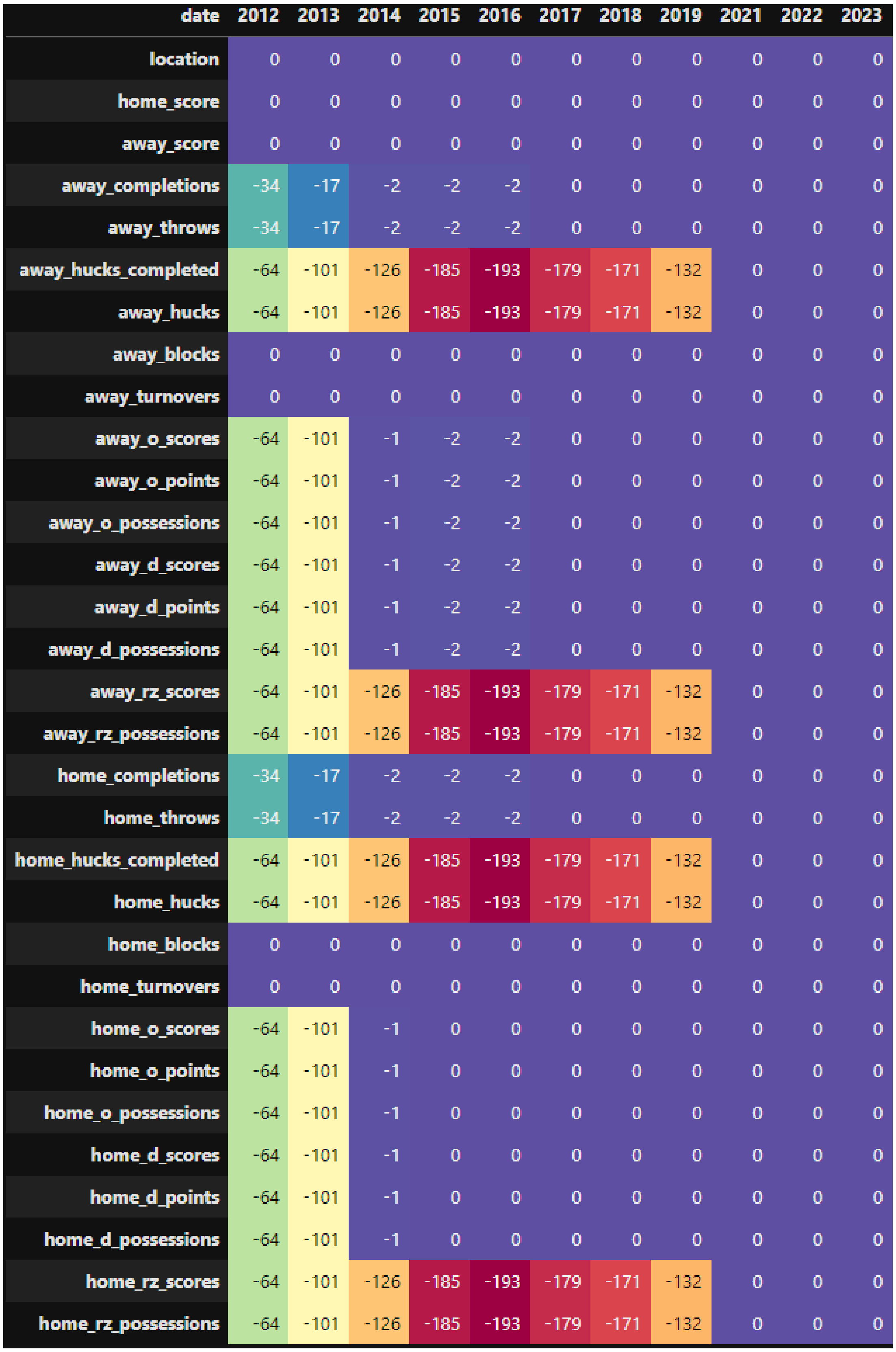


date	2012	2013	2014	2015	2016
home_score	0	0	0	0	0
away_score	0	0	0	0	0
home_completions	-34	-17	-2	-2	-2
away_completions	-34	-17	-2	-2	-2
home_throws	-34	-17	-2	-2	-2
away_throws	-34	-17	-2	-2	-2
home_blocks	0	0	0	0	0
away_blocks	0	0	0	0	0
home_turnovers	0	0	0	0	0
away_turnovers	0	0	0	0	0

Final Note

As mentioned in Appendix I – Statistics Glossary, there is an imbalance between some basic stats. One idea that I did not try for this project was to add opposing blocks to a team's throws. It is possible that a team's turnover is counted as both an incomplete pass and a turnover, but an opponent block may not count towards an incomplete pass.

Missing Data (missing values set to -1 so actual 0 values don't count as missing)



date	2012	2013	2014	2015	2016	2017	2018	2019	2021	2022	2023
location	0	0	0	0	0	0	0	0	0	0	0
home_score	0	0	0	0	0	0	0	0	0	0	0
away_score	0	0	0	0	0	0	0	0	0	0	0
away_completions	-34	-17	-2	-2	-2	0	0	0	0	0	0
away_throws	-34	-17	-2	-2	-2	0	0	0	0	0	0
away_hucks_completed	-64	-101	-126	-185	-193	-179	-171	-132	0	0	0
away_hucks	-64	-101	-126	-185	-193	-179	-171	-132	0	0	0
away_blocks	0	0	0	0	0	0	0	0	0	0	0
away_turnovers	0	0	0	0	0	0	0	0	0	0	0
away_o_scores	-64	-101	-1	-2	-2	0	0	0	0	0	0
away_o_points	-64	-101	-1	-2	-2	0	0	0	0	0	0
away_o_possessions	-64	-101	-1	-2	-2	0	0	0	0	0	0
away_d_scores	-64	-101	-1	-2	-2	0	0	0	0	0	0
away_d_points	-64	-101	-1	-2	-2	0	0	0	0	0	0
away_d_possessions	-64	-101	-1	-2	-2	0	0	0	0	0	0
away_rz_scores	-64	-101	-126	-185	-193	-179	-171	-132	0	0	0
away_rz_possessions	-64	-101	-126	-185	-193	-179	-171	-132	0	0	0
home_completions	-34	-17	-2	-2	-2	0	0	0	0	0	0
home_throws	-34	-17	-2	-2	-2	0	0	0	0	0	0
home_hucks_completed	-64	-101	-126	-185	-193	-179	-171	-132	0	0	0
home_hucks	-64	-101	-126	-185	-193	-179	-171	-132	0	0	0
home_blocks	0	0	0	0	0	0	0	0	0	0	0
home_turnovers	0	0	0	0	0	0	0	0	0	0	0
home_o_scores	-64	-101	-1	0	0	0	0	0	0	0	0
home_o_points	-64	-101	-1	0	0	0	0	0	0	0	0
home_o_possessions	-64	-101	-1	0	0	0	0	0	0	0	0
home_d_scores	-64	-101	-1	0	0	0	0	0	0	0	0
home_d_points	-64	-101	-1	0	0	0	0	0	0	0	0
home_d_possessions	-64	-101	-1	0	0	0	0	0	0	0	0
home_rz_scores	-64	-101	-126	-185	-193	-179	-171	-132	0	0	0
home_rz_possessions	-64	-101	-126	-185	-193	-179	-171	-132	0	0	0