U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

PREDICTION PROJECT

BASKET BALL

Diogo Neves – up202108460 – IF : 1
Pedro Paixão – up202008467 – IF : 1
Tiago Torrado – up202206399 – IF : 1

Group 76

OBJECTIVES

Business Objective

- Support pre-season decision-making using data-driven predictions
- All predictions are made before the season starts, without using any games from the target year

Project Objectives

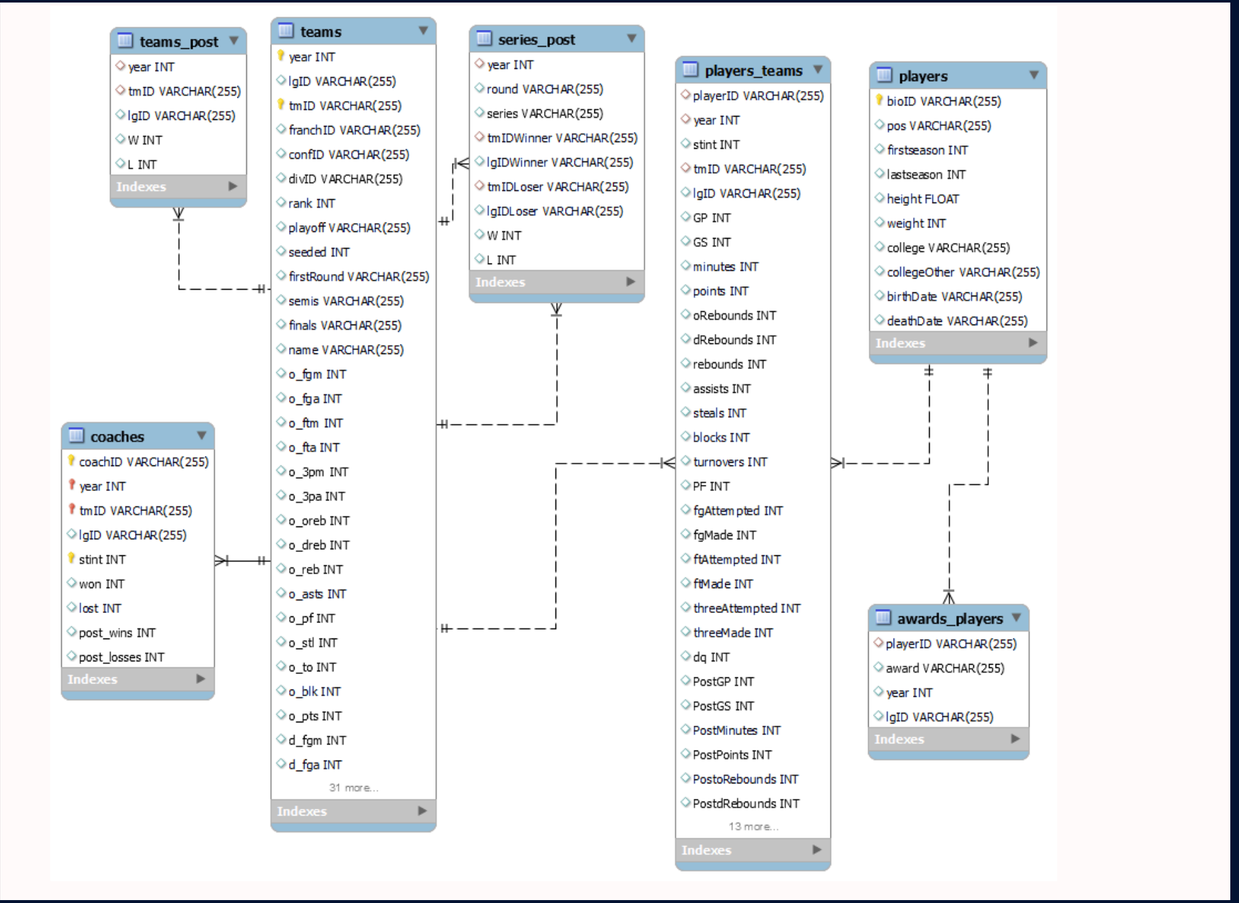
- Predict final conference rankings using historical team-level data
- Identify teams likely to change head coaches based on past performance trends
- Predict individual award winners using player performance metrics

Modeling Strategy

- Each objective is addressed with a task-specific machine learning approach, reflecting different data structures, targets, and evaluation criteria

DATASET

- **awards_players** (96 records) – Contains all awards and honors received by players over the 10 seasons.
- **coaches** (163 records) – Lists all coaches who managed teams during the covered period.
- **players** (894 records) – Provides detailed information about all players who participated in the WNBA throughout these seasons.
- **players_teams** (1877 records) – Describes each player's performance for every team they played for.
- **series_post** (71 records) – Contains the results of postseason series.
- **teams** (143 records) – Includes the regular-season performance statistics for each team.
- **teams_post** (81 records) – Describes the postseason results for each team.



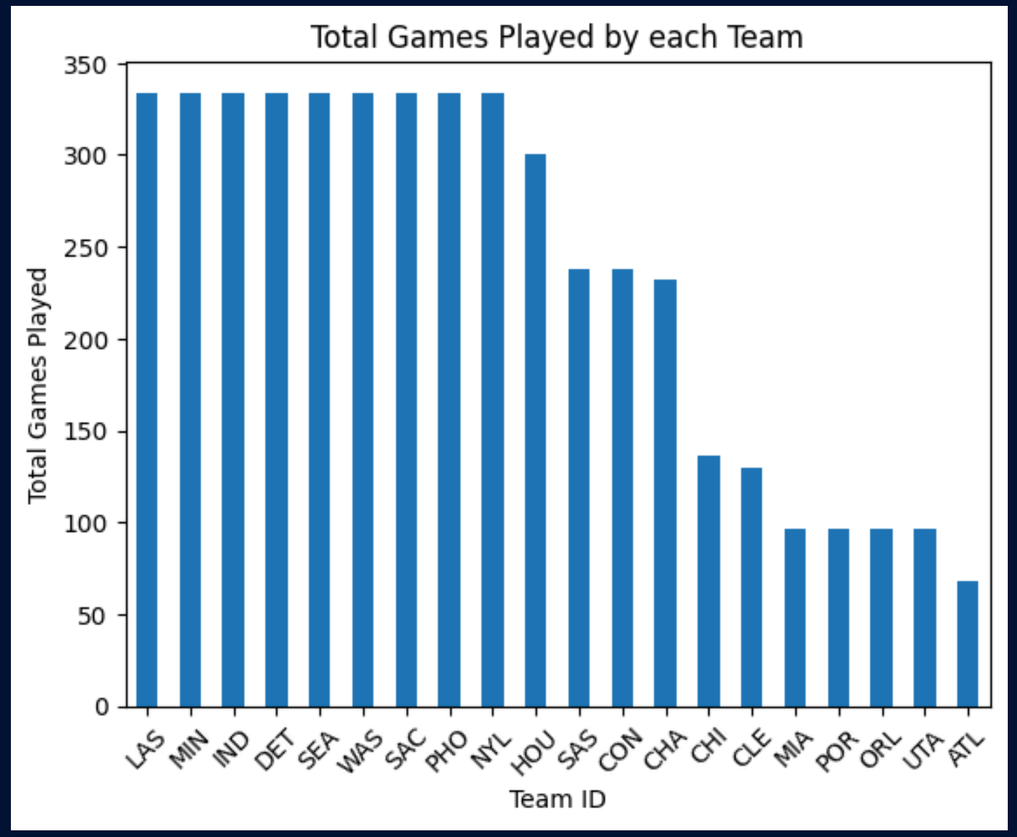
EDA

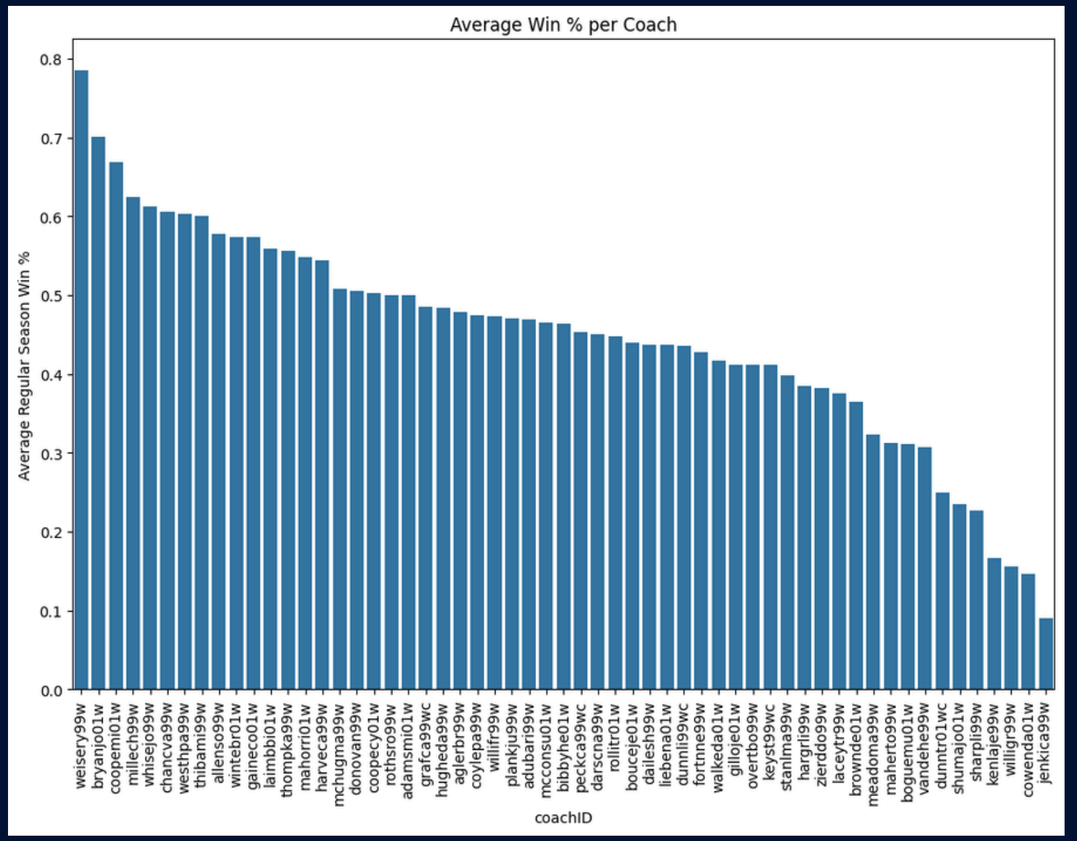
Exploratory Data Analysis

We generated several plots focusing on identifying correlations amongst various features:

- Total Games Played by teams
- Average Win Ratio per Coach
- Effects of Previous Season Performance on Final Ranking
- Relationship between Height and Weight
- Relationship between features ...

EDA





The chart reveals a significant disparity in historical records, with established franchises having 3x more data points than newer teams.

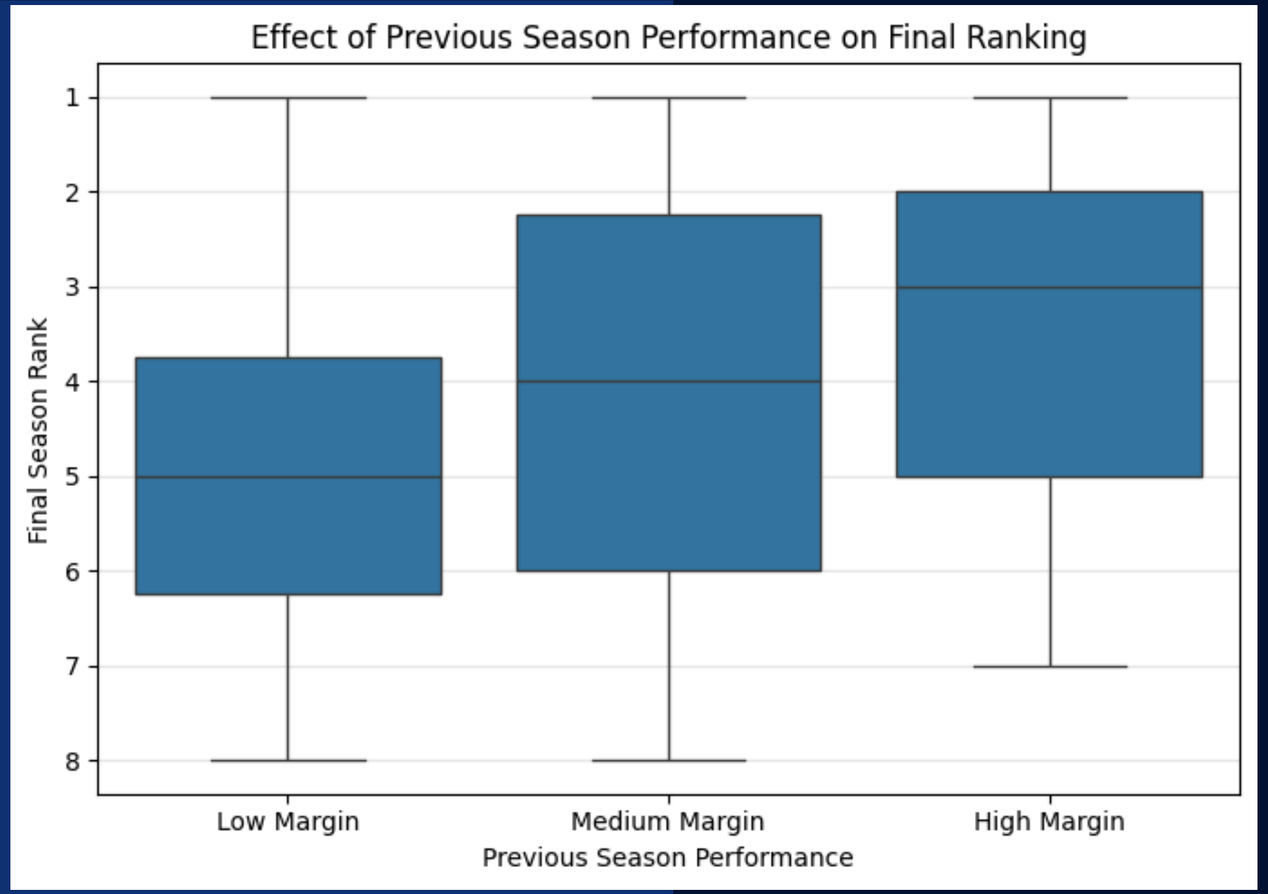
The chart displays the stark contrast in coaching efficiency, ranging from dominant tenures (80% win rate) to struggling stints (<45%), serving as the primary feature for assessing job security.

EDA

The boxplot shows the relationship between **last season's performance margin** (low, medium, high) and **next season's final ranking**:

- Trend: **Teams with high previous margins tend to finish higher** (lower ranking numbers), while low-margin teams have more scattered, worse results.
- Median Rankings: Low ~5, Medium ~4, High ~3.
- Variability: Low and Medium-margin teams are less consistent; High-margin teams are more stable.

Conclusion: **Previous season performance is a good predictor of next season's ranking**—strong teams tend to stay strong, weaker teams show more variation.



EDA

Issues Observed in the Height vs Weight Data (Before Imputation):

Zero Values:

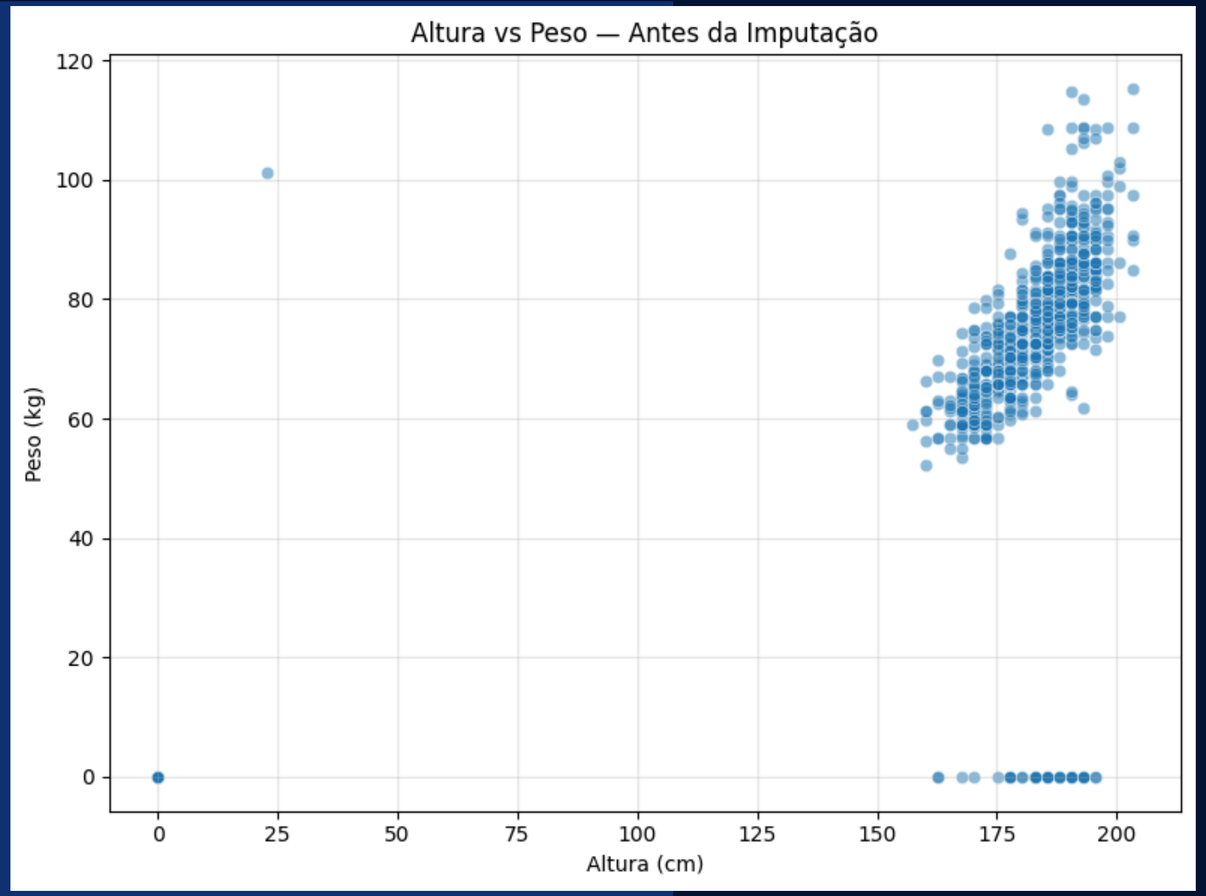
- Several players have height or weight recorded as 0.
- These are invalid entries and distort the analysis.

Extreme Outliers:

- Some data points are unrealistically low or high (e.g., ~25 cm height with ~100 kg weight).
- Likely data entry errors or conversion mistakes.

Missing Values Indicated by Zeros:

- Many players have weight = 0 even when height is valid.
- These need to be imputed for meaningful analysis.



DATA PREPARATION

teams

- Created a working copy of the original teams_df for final cleaning steps.
- Removed irrelevant columns such as **divID** and seeded to simplify the dataset.

players

- Converted height/weight to numeric and **removed impossible values**.
- Imputed **missing values using regression**.
- **Removed outliers** and **converted units** (cm/kg).
- Standardized positions and college fields, filling missing entries with “Unknown”.
- Dropped redundant or low-value columns.

coaches

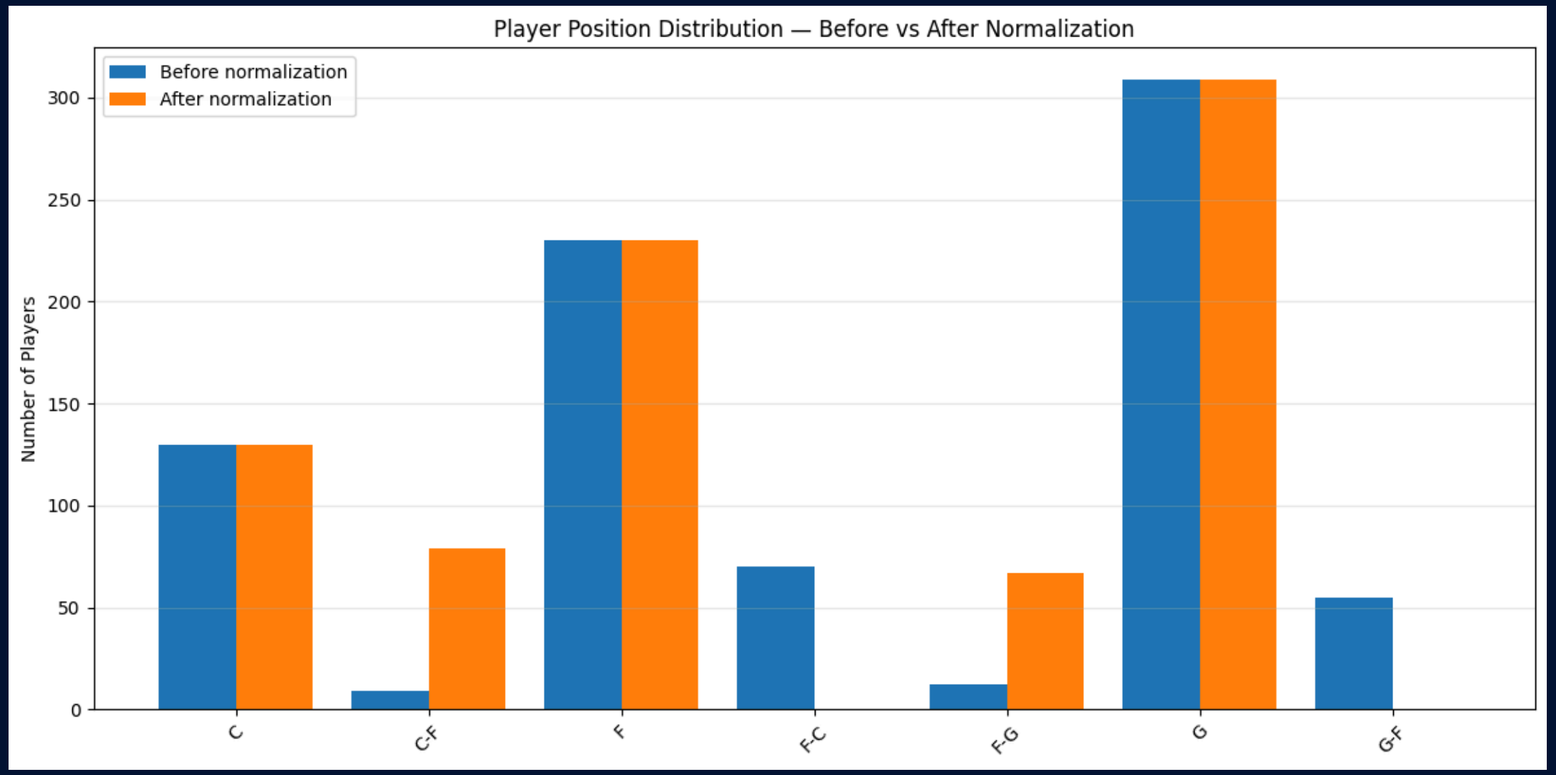
- Identified and kept only the coaches who started the season (Minimum Stint) for each team, discarding interim coaches to focus predictions on the primary head coach.

Other tables

For the remaining datasets, attributes considered irrelevant (e.g., lgID) or columns with only a single unique value were discarded.

DATA PREPARATION

Player positions **were normalized by merging equivalent combinations** (e.g., F–C and C–F), reducing redundant categories and improving data consistency.



DATA PREPARATION

Data Cleaning: Height & Weight Imputation

Objective :

- Fix missing or invalid (zero) values
- Keep all players in the dataset

Method :

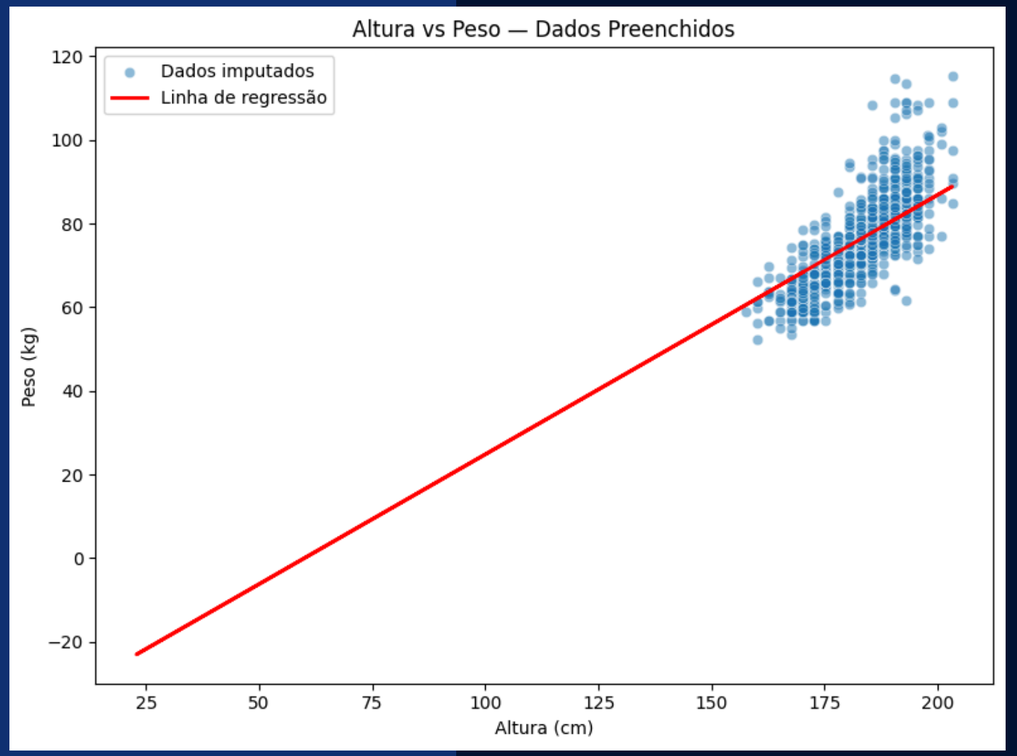
- Used only valid records (height > 0, weight > 0)
- Learned the relationship between height and weight using linear regression

Imputation :

- Missing weight → predicted from height
- Missing height → predicted from weight

Post-Processing :

- Converted units (in → cm, lb → kg)
- Checked distributions and confirmed no remaining zeros



Feature Enginnering

RANKING OF THE REGULAR SEASON

```python
agg["FG%"] = np.where(agg["fgAttempted"] > 0, agg["fgMade"] / agg["fgAttempted"], 0)
agg["3P%"] = np.where(agg["threeAttempted"] > 0, agg["threeMade"] / agg["threeAttempted"], 0)
agg["FT%"] = np.where(agg["ftAttempted"] > 0, agg["ftMade"] / agg["ftAttempted"], 0)

# Weighted Efficiency Index (estilo PER)
agg["player_rating"] = (
    (agg["points"])
    + (agg["rebounds"] * 1.2)
    + (agg["assists"] * 1.5)
    + (agg["steals"] * 3.0)
    + (agg["blocks"] * 3.0)
    + (agg["threeMade"] * 0.8)
    - (agg["turnovers"] * 2.0)
    - (agg["PF"] * 0.5)
) / (agg["minutes"] / 36 + 1)

agg["eff_bonus"] = 0.5 * agg["FG%"] + 0.3 * agg["FT%"] + 0.2 * agg["3P%"]
agg["player_rating"] *= (1 + agg["eff_bonus"])

agg = agg[(agg["minutes"] >= 100) & (agg["minutes"] < agg["minutes"].quantile(0.99))].copy()
```

- **margin_prev** – Average point differential per game in the previous season (offense – defense).
- **avg_player_rating** – Weighted average performance rating of all roster players before the season starts.
- **team_total_awards** – Total number of career awards accumulated by players on the roster.
- **elite_ratio_prev** – Proportion of roster players ranked in the top 10% based on historical performance
- All features are computed using information available **prior to the season** (past year information only) **to avoid data leakage**.

The model calculates a score for each team based on **individual talent**, **awards**, and **past performance**, normalizes these metrics, and then ranks the teams within each conference to generate the predicted ranking. All main features are normalized using **z-score** to make them comparable and to control for outliers.

```python
# Histórico da equipa (T-1)
teams = teams.sort_values(["tmID", "year"]).copy()
for col in ["rank", "won", "lost", "GP", "o_pts", "d_pts"]:
    teams[f"{col}_prev"] = teams.groupby("tmID")[col].shift(1)

teams["win_pct_prev"] = teams["won_prev"] / teams["GP_prev"].replace(0, np.nan)
teams["margin_prev"] = (teams["o_pts_prev"] - teams["d_pts_prev"]) / teams["GP_prev"].replace(0, np.nan)

data_ready = teams.merge(team_sum, on=["tmID", "year"], how="left")
```

$$X_{\text{norm}} = \frac{X - \text{média}}{\text{desvio padrão}}$$

EVALUATION

Evaluation Objective

Ensure that the model can predict the final ranking of a season using only historical information from the previous year (T–1).

Methodology - Walk-Forward Evaluation

For each year Y:

1. Train the model using all seasons prior to Y (< Y).
2. Evaluate performance on season Y.
3. Repeat this process for all available years.

How the Prediction Is Made

- The model outputs a continuous score: pred_raw
- The final ranking is obtained by sorting teams within each conference
- → pred_rank = rank(pred_raw, ascending=True)

Evaluation Metrics

- MAE - Mean Absolute Error
- Spearman Correlation
- Spearman per Conference
- Variance of the Spearman correlations

```python
X_COLS = [
    "margin_prev",
    "avg_player_rating",
    "team_total_awards",
    "elite_ratio_prev",
]

def evaluate_year(test_year):
    train = data[data["year"] < test_year]
    test = data[data["year"] == test_year]

    if len(train) == 0 or len(test) == 0:
        return None

    X_train = train[X_COLS]
    y_train = train["rank"].astype(float)

    X_test = test[X_COLS]
    y_test = test["rank"].astype(float)

    model = LinearRegression()
    model.fit(X_train, y_train)

    test["pred_raw"] = model.predict(X_test)

    test["pred_rank"] = test.groupby("confID")["pred_raw"].rank(ascending=True, method="min")

    mae = mean_absolute_error(y_test, test["pred_rank"])
    spearman, _ = spearmanr(y_test, test["pred_rank"])

    preds = test[["tmID", "year", "confID", "rank", "pred_rank"]].copy()

    return {"year": test_year, "MAE": mae, "Spearman": spearman, "preds": preds}
```

MODEL SELECTION

We initially trained a **single global model** using **both conferences together**. Early analysis suggested a pattern:

- **EA** performed slightly better with **Random Forest** (RF).
- **WE** responded better to **Linear Regression** (LR).

To investigate, we trained separate models for each conference (RF for EA, LR for WE).

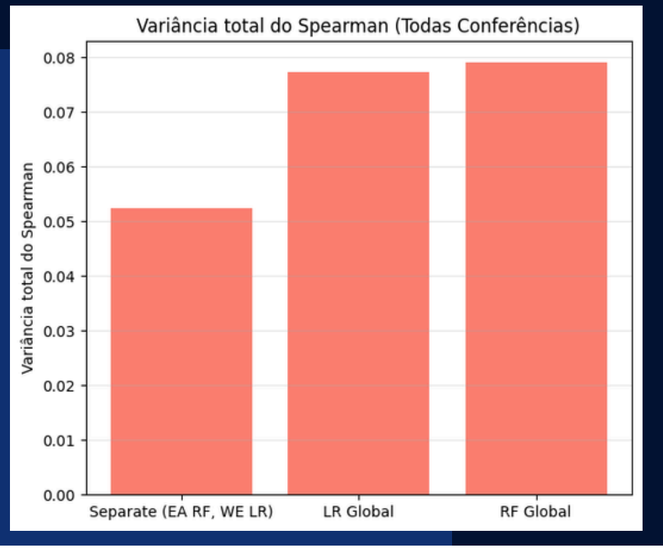
While the global LR model achieved higher mean Spearman, the conference-specific models offered a key advantage:

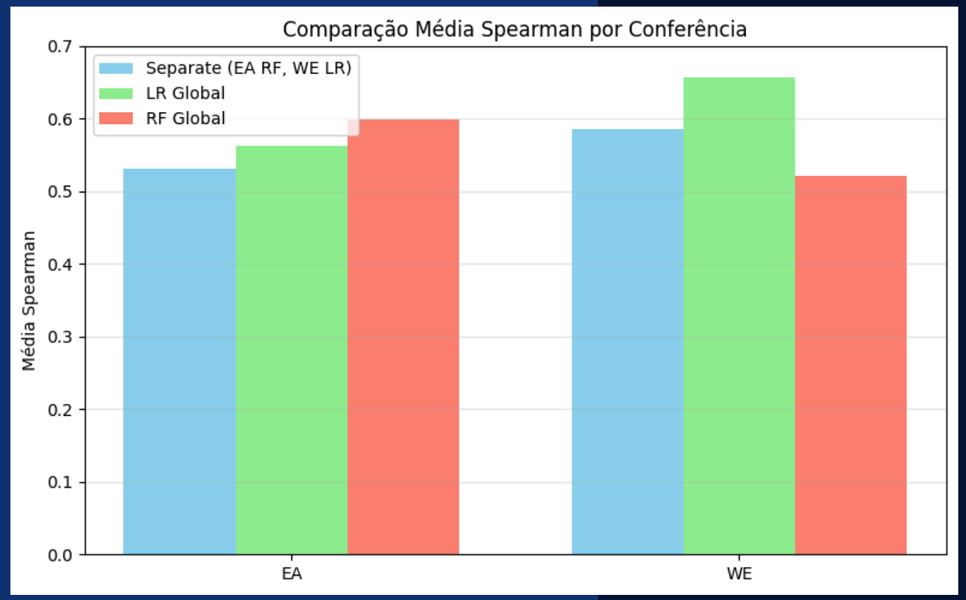
- They produced the **lowest variance** across years, ensuring **more stable** and **consistent predictions** from season to season.

Although **Random Forest** performs slightly better for EA , its overall variance is higher and predictions are less stable compared to the conference-separated models and the Linear Regression model.

Since our priority is **predictive stability over peak accuracy**, we ultimately chose the **conference-separated models**, even though:

- They do not reach the highest Spearman,
- They are slightly weaker in raw performance,
- Yet they provide the **most stable** and **low-variance forecasts**, which is crucial for long-term predictions.



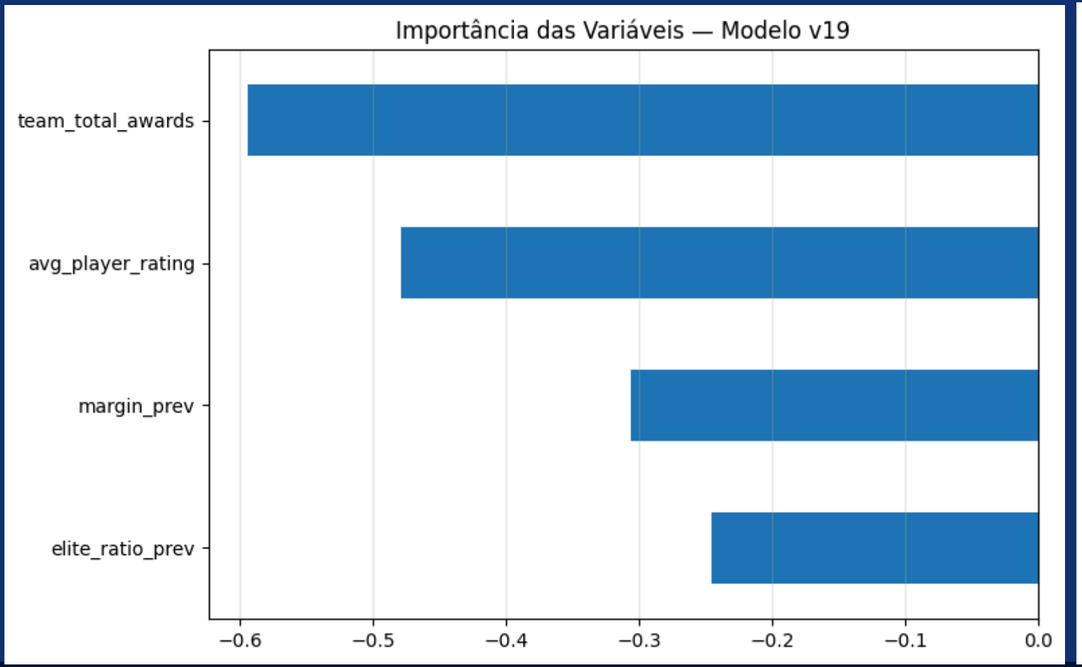


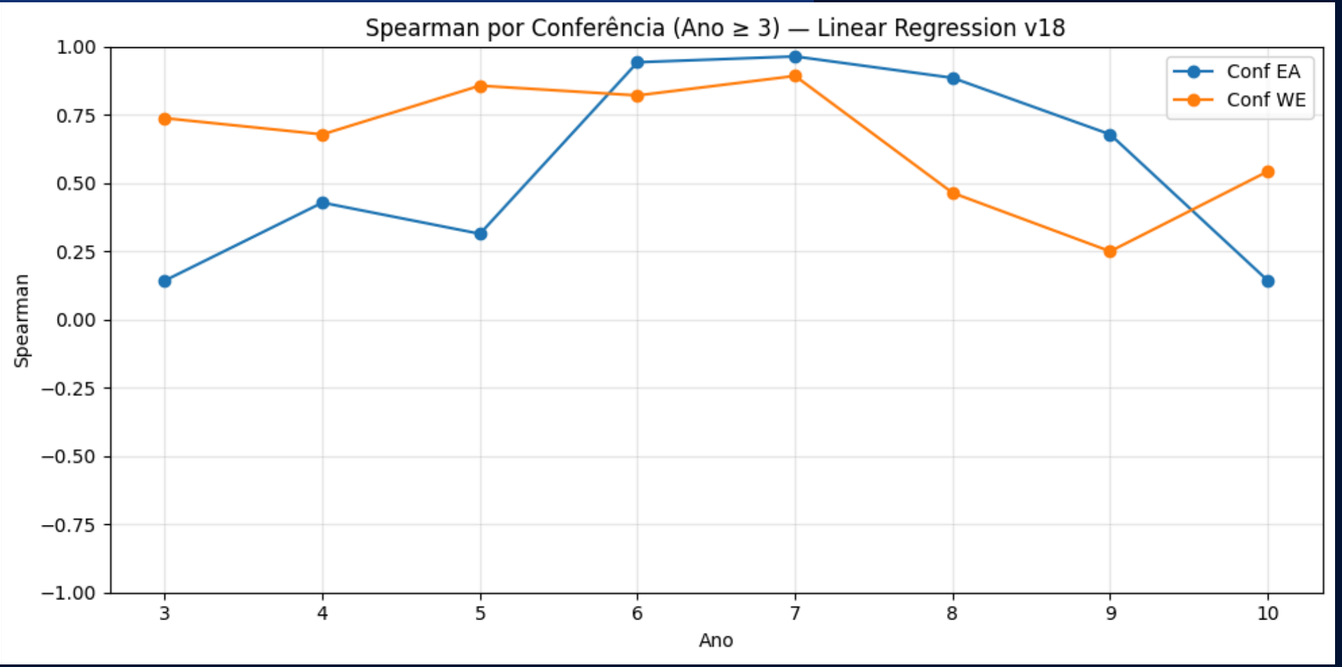
RESULTS - LR

Achieved the **best overall Spearman correlation across both conferences**.
Particularly strong in the Western Conference, outperforming Random Forest.
Simple, interpretable, and generalizes well on small, season-level datasets.

```
📈 Média MAE: 1.261
📈 Média Spearman: 0.610
```

```
📌 --- Estatísticas Globais dos Spearmans (todas as conferências) ---
Total de valores usados: 16
Variância total (todas confs): 0.07725
Desvio padrão total (todas confs): 0.27795
----------------------------------------------------------------------
```





RESULTS- RF

Why use Random Forest ?

- Captures non-linear relationships better than Linear Regression.
- Handles interactions between features automatically.
- More stable than a single decision tree due to averaging.

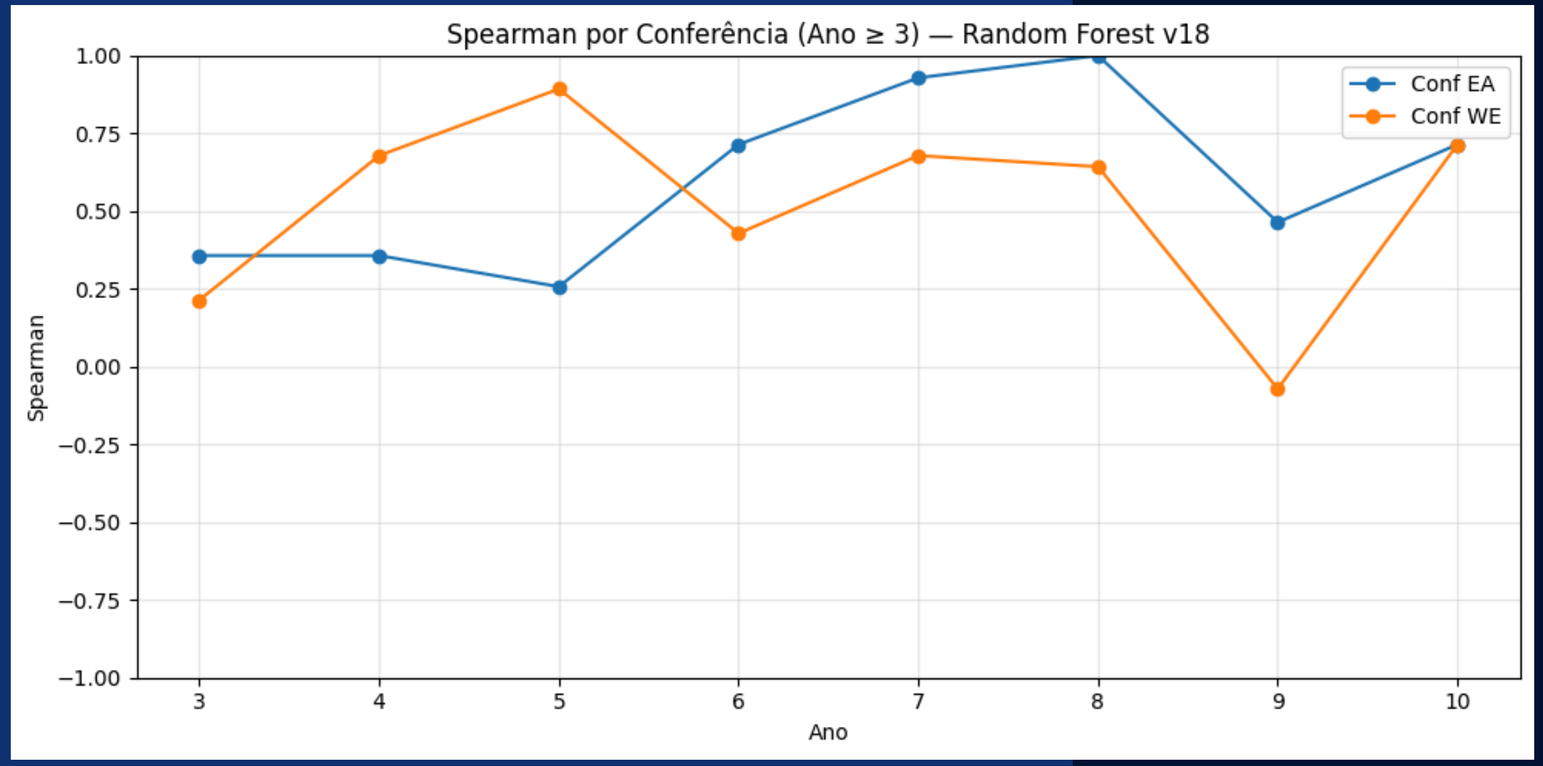
Performance Summary

- Best-performing model for EA Conference.
- Its overall variance is higher compared to the other models.
- Slightly weaker than LR in WE, suggesting different pattern complexity.

Why keep it ?

- Provides complementary strengths to LR.
- Useful when prediction patterns vary between conferences.

Mean Spearman – 0.559



📌 Número total de Spearmans usados = 16
📌 Variância de TODOS os Spearmans = 0.07908
📌 Desvio padrão de TODOS os Spearmans = 0.28121

RESULTS - COMBINED

Why use conference-specific models?

- Captures differences in predictive patterns between conferences.
- EA tends to benefit from **Random Forest**; WE performs better with **Linear Regression**.

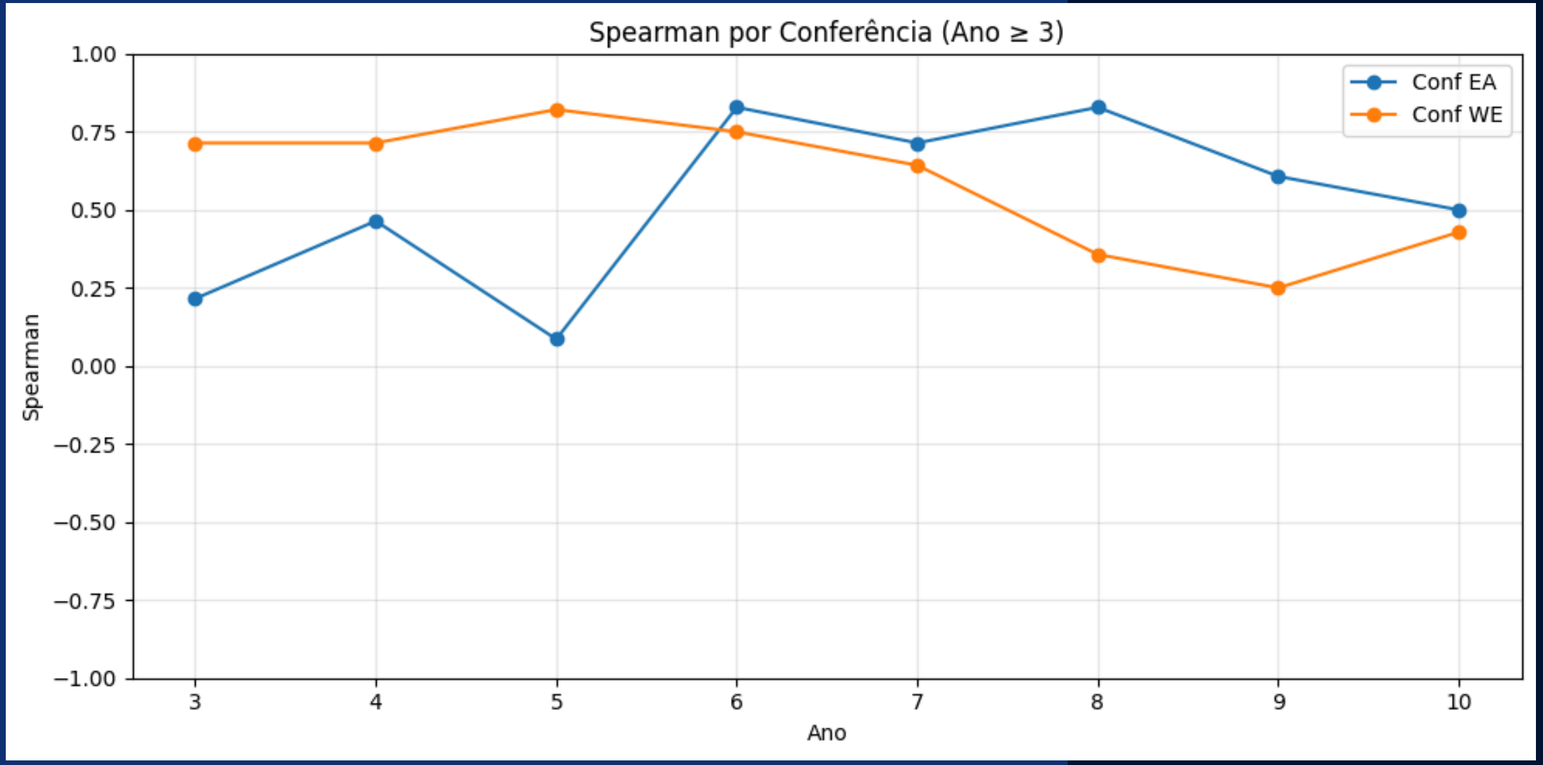
Performance Summary

- EA (RF): lower mean Spearman than global RF and LR but lowest variance → more stable predictions.
- WE (LR): consistently strong Spearman, also low variance.
- Global LR achieves higher Spearman overall but with higher variance → less consistency across seasons.

Why keep them?

- Provides the most stable and reliable forecasts for long-term planning.

Mean Spearman – 0.557



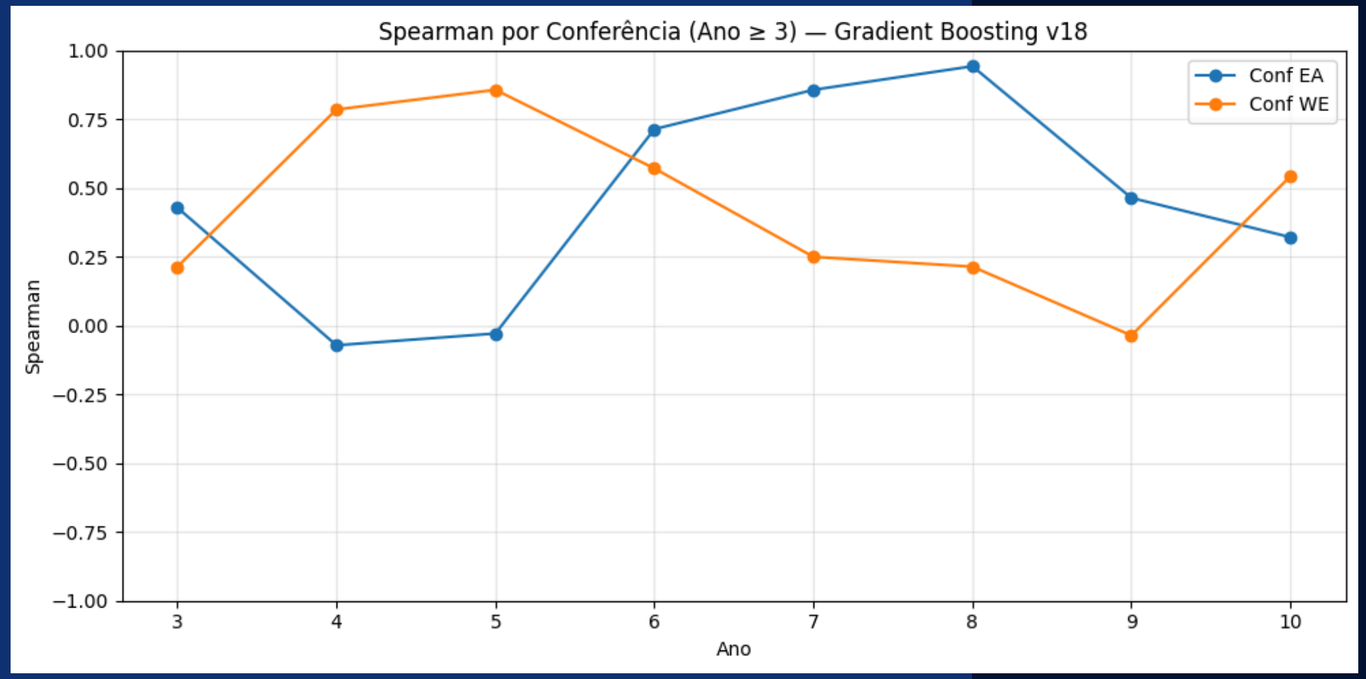

```
📌 Spearman geral (todas conferências, Ano ≥ 3):
Média: 0.5577
Variância: 0.05250
```

RESULTS - GB

Mean Spearman – 0.436

Why XGBoost Performed Worst Overall

- XGBoost is a high-capacity model requiring large amounts of data.
- With few seasons and one-year lag features (T–1), the model overfits to noise instead of general patterns.
- Complex tree-based models like XGBoost do not gain much from these linear relationships and can even overcomplicate predictions.
- However, XGBoost achieved slightly better results on the **EA conference**, which suggests that for EA, **tree-based models** such as **Random Forest** might capture non-linear patterns more effectively than linear models.



📌 Número total de Spearmans usados = 16
📌 Variância de TODOS os Spearmans (GBM) = 0.10523
📌 Desvio padrão de TODOS os Spearmans (GBM) = 0.32439

RESULTS - TEST SEASON 11

Test Season Prediction - Conference-separated Models

- Eastern Conference (EA): **Random Forest** used to capture non-linear team patterns.
- Western Conference (WE): **Linear Regression** used due to stable, mostly linear feature relationships.

ac-data-mining-project > a > predictions_test_v19_by_conf.csv > data

```
1   tmID,confID,year,rank_pred
2   ATL,EA,11,1.0
3   WAS,EA,11,2.0
4   IND,EA,11,3.0
5   CON,EA,11,4.0
6   CHI,EA,11,5.0
7   NYL,EA,11,6.0
8   PHO,WE,11,1.0
9   SAS,WE,11,2.0
10  MIN,WE,11,3.0
11  SEA,WE,11,4.0
12  LAS,WE,11,5.0
13  TUL,WE,11,6.0
```

COACH CHANGES PREDICTIONS

```
features = [
    "expectation_gap",
    "coach_award_count",
    "playoff_miss_streak",
    "prev_underperformance",
    "tenure",
    "momentum"
]
```

Features :

- **expectation_gap** – Checks if the coach is currently performing worse than the team's 3-year historical win ratio weighted average
- **coach_award_count** - Total "Coach of the Year" awards won in the past
- **playoff_miss_streak** - Number of consecutive years that the team is missing the playoffs
- **prev_underperformance** - Checks if the team won fewer games last year than their stats would suggest
- **tenure** - Years with the current team
- **momentum** - The change in Win percentage from two years ago vs. last year.

EVALUATION

Evaluation Objective

Ensure that the model can predict mid-season coach changes before the season starts, using only historical information from the previous year (T–1).

Methodology — Walk-Forward Evaluation

For each year Y:

1. Train the model using all seasons prior to Y (< Y).
2. Evaluate performance on season Y.
3. Repeat this process for all available years.

How the Prediction Is Made

- Target: mid_season_change (Binary: 1 = Fired, 0 = Safe).
- Model Architecture: A Soft Voting Ensemble:
  - Logistic Regression ("The Alarmist"): Aggressively weighted to catch rare firing signals.
  - Random Forest ("The Shield"): Conservative filter to reduce false alarms.
  - Gradient Boosting ("The Learner"): Adds stability to the average.
- Final_Probability = Average(LR_Prob, RF_Prob, GB_Prob)
- If Final_Probability > 0.45 → Add Team to "Change Set".

Evaluation Metrics

- AUC — Area Under Curve
- F1 Score

RESULTS - LR

- The model acts as a High-Sensitivity Risk Detector.
- It is extremely aggressive, consistently flagging a large number of teams as "at risk."
- While this ensures it rarely misses a firing (High Recall), it generates many "False Alarms" (Low Precision), predicting changes that never happen.

```
Average Metrics (Years 3-10):
AUC           0.452134
F1            0.215647
Precision     0.133239
Recall        0.666667
Spearman     -0.021018
Accuracy      0.332246
```

```
                                Predicted_Teams_Set
0  {CLE, NYL, WAS, SAC, CHA, IND, POR, LAS, MIA, ...
1            {MIN, SAS, CLE, NYL, SAC, CON, CHA, PHO}
2            {NYL, WAS, HOU, CON, CHA, IND, PHO, SEA}
3  {NYL, WAS, HOU, SAC, CON, CHA, IND, LAS, DET, ...
4  {MIN, NYL, WAS, HOU, CON, CHA, CHI, IND, LAS, ...
5  {MIN, NYL, WAS, HOU, SAC, CON, IND, CHI, LAS, ...
6  {NYL, HOU, SAC, CON, ATL, IND, PHO, LAS, DET, ...
7  {NYL, WAS, SAC, ATL, CHI, IND, PHO, LAS, DET, ...
```

Detailed Results:

	Year	AUC	F1	Precision	Recall	Spearman	Accuracy
0	3	0.102564	0.000000	0.000000	0.000000	-0.538418	0.125000
1	4	0.708333	0.400000	0.250000	1.000000	0.253185	0.571429
2	5	0.400000	0.181818	0.125000	0.333333	-0.146385	0.307692
3	6	0.681818	0.333333	0.200000	1.000000	0.227921	0.384615
4	7	0.307692	0.166667	0.090909	1.000000	-0.172005	0.285714
5	8	0.416667	0.181818	0.100000	1.000000	-0.077152	0.307692
6	9	0.000000	0.000000	0.000000	0.000000	-0.447214	0.214286
7	10	1.000000	0.461538	0.300000	1.000000	0.731925	0.461538

RESULTS - LR

Mean AUC – 0.452

Mean F1 – 0.215

RESULTS - RF

- The model acts as a Conservative Stabilizer.
- It is more cautious than LR, prioritizing stability over sensitivity by filtering out noise and complex non-linear patterns.
- While this ensures that its predictions are more trustworthy (High Precision), it tends to be too hesitant (Low Recall).

Average Metrics (Years 3-10):

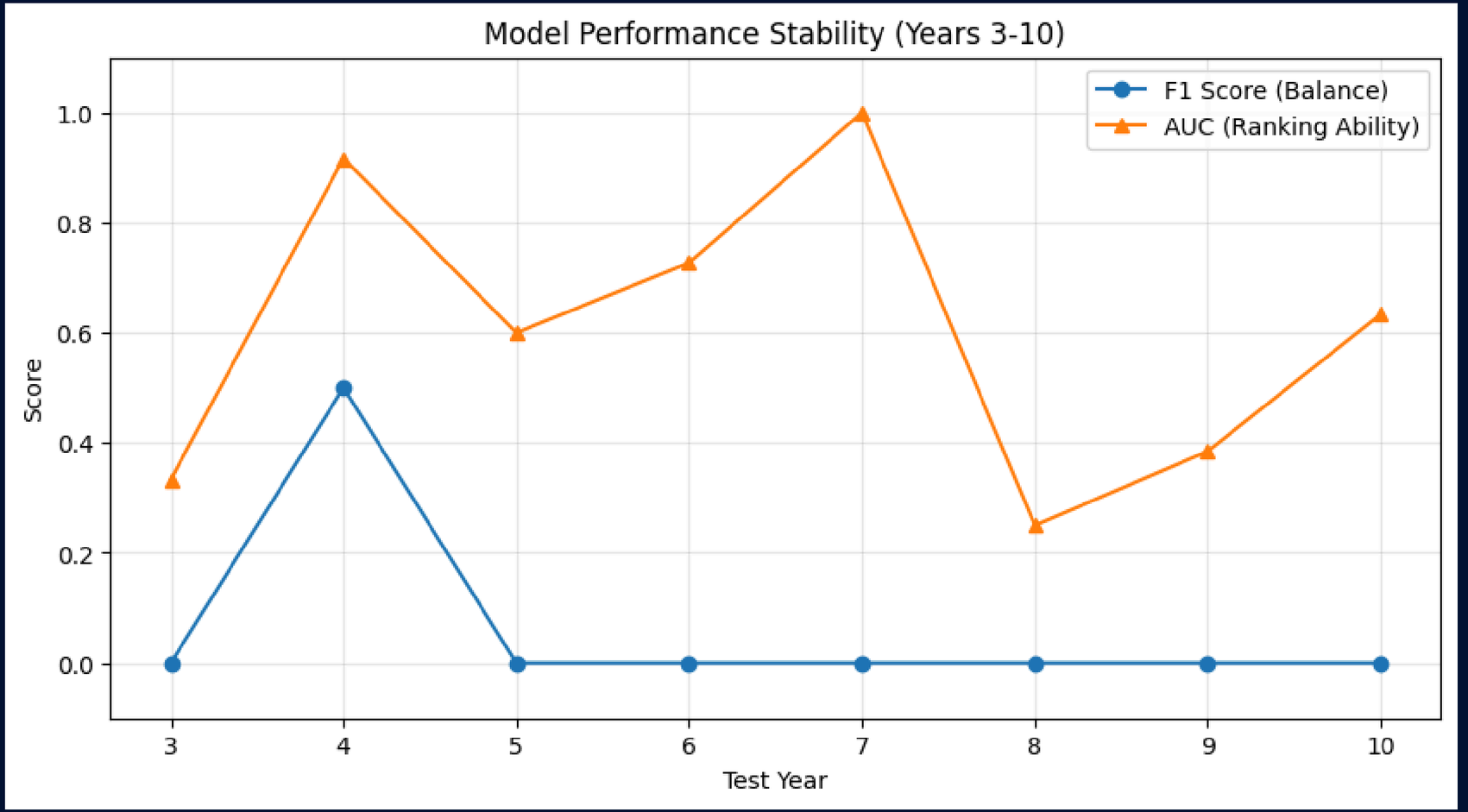
AUC	0.605653
F1	0.062500
Precision	0.062500
Recall	0.062500
Spearman	0.127430
Accuracy	0.844694

Detailed Results:

	Year	AUC	F1	Precision	Recall	Spearman	Accuracy
0	3	0.333333	0.0	0.0	0.0	-0.225955	0.812500
1	4	0.916667	0.5	0.5	0.5	0.506370	0.857143
2	5	0.600000	0.0	0.0	0.0	0.146385	0.769231
3	6	0.727273	0.0	0.0	0.0	0.284901	0.846154
4	7	1.000000	0.0	0.0	0.0	0.447214	0.928571
5	8	0.250000	0.0	0.0	0.0	-0.231455	0.846154
6	9	0.384615	0.0	0.0	0.0	-0.103203	0.928571
7	10	0.633333	0.0	0.0	0.0	0.195180	0.769231

	Predicted_Teams_Set
0	{}
1	{PHO, SAC}
2	{}
3	{}
4	{}
5	{NYL}
6	{}
7	{}

RESULTS - RF



Mean AUC – 0.605

Mean F1 – 0.062

RESULTS - GB

- The model acts as an Adaptive Learner.
- It operates differently from the others by training sequentially, building new trees specifically to fix the errors made by previous ones.
- This makes it the "Middle Ground" of the ensemble: it is more precise than the Logistic Regression but more flexible than the Random Forest, effectively acting as the tie-breaker when the other two disagree.

Average Metrics (Years 3-10):

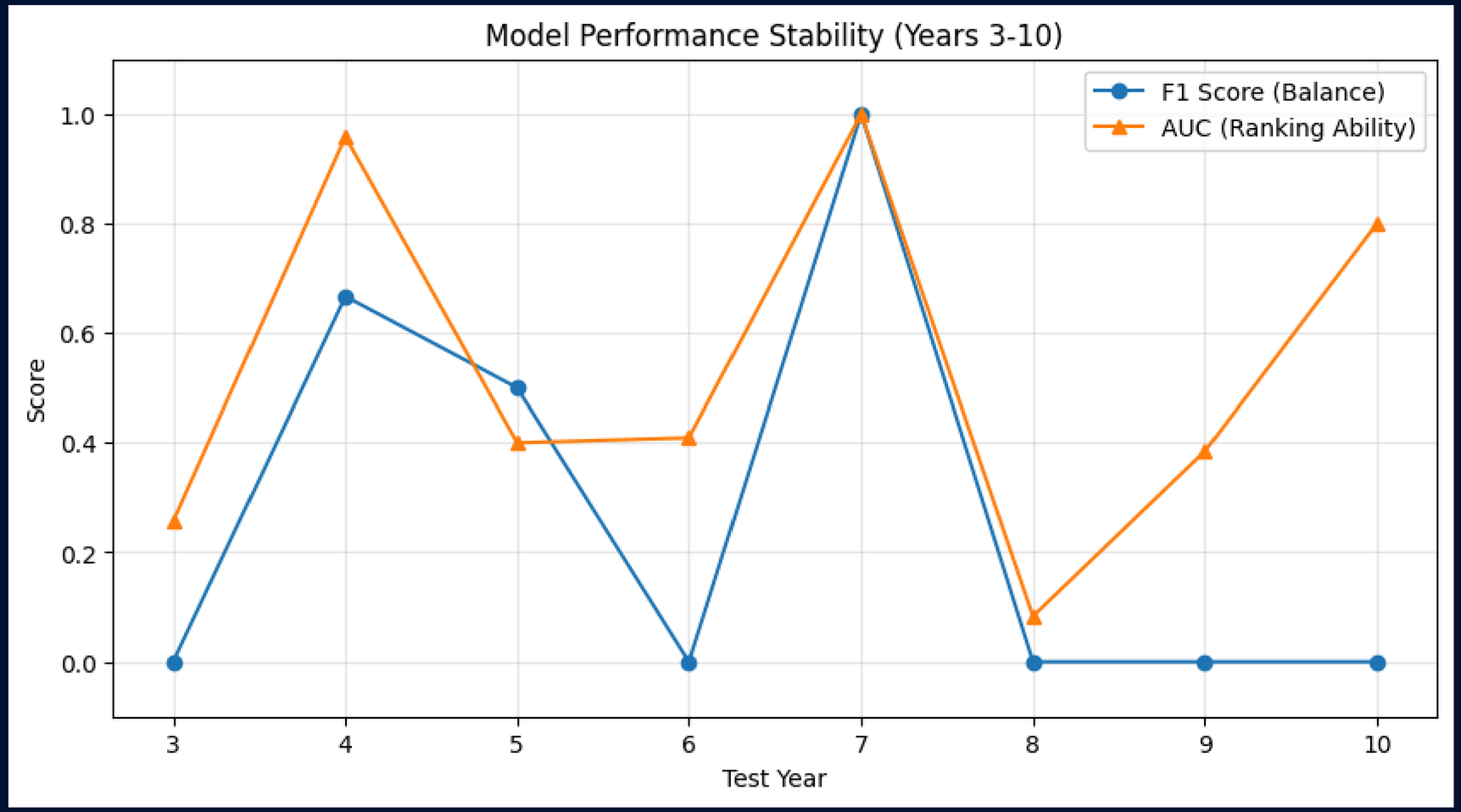
AUC	0.536473
F1	0.270833
Precision	0.375000
Recall	0.229167
Spearman	0.045844
Accuracy	0.829499

Detailed Results:

	Year	AUC	F1	Precision	Recall	Spearman	Accuracy
0	3	0.256410	0.000000	0.0	0.000000	-0.330241	0.625000
1	4	0.958333	0.666667	1.0	0.500000	0.557007	0.928571
2	5	0.400000	0.500000	1.0	0.333333	-0.146385	0.846154
3	6	0.409091	0.000000	0.0	0.000000	-0.113961	0.692308
4	7	1.000000	1.000000	1.0	1.000000	0.447706	1.000000
5	8	0.083333	0.000000	0.0	0.000000	-0.385758	0.846154
6	9	0.384615	0.000000	0.0	0.000000	-0.103203	0.928571
7	10	0.800000	0.000000	0.0	0.000000	0.441588	0.769231

	Predicted_Teams_Set
0	{UTA, NYL, WAS}
1	{SAC}
2	{LAS}
3	{PHO, DET}
4	{MIN}
5	{NYL}
6	{}
7	{}

RESULTS - GB



Mean AUC – 0.536

Mean F1 – 0.270

RESULTS VOTING-ENSEMBLE

- Instead of relying on a single opinion, the model requires a consensus among the three models (LR, RF, GB) before flagging a team.
- It effectively reduces the false alarms of LR, while still giving decent to results in predicting coaches changes.
- If all three models see moderate risk (50%, 45%, 45%), the average (≈ 47%) triggers a flag. However, if one model predicts high risk (80%) and the other two a low risk (20%, 20%), the combined average (≈ 40%) will not be high enough to trigger a flag.
- We believe this “triangulation” approach is a great way to predict coach changes.

Average Metrics (Years 3-10):

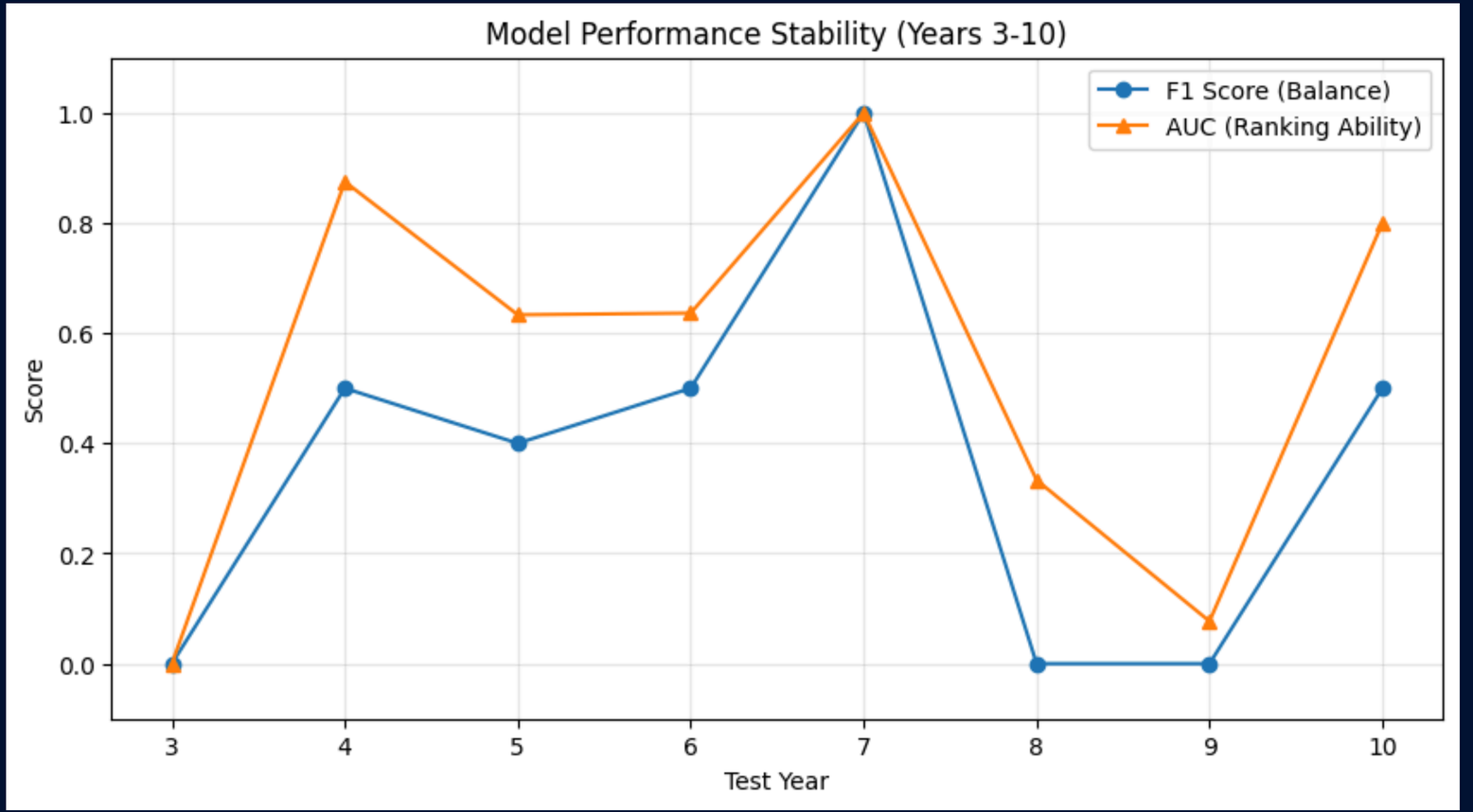
AUC	0.544369
F1	0.362500
Precision	0.437500
Recall	0.333333
Spearman	0.062268
Accuracy	0.805203

Detailed Results:

	Year	AUC	F1	Precision	Recall	Spearman	Accuracy
0	3	0.000000	0.0	0.0	0.000000	-0.677365	0.562500
1	4	0.875000	0.5	0.5	0.500000	0.455733	0.857143
2	5	0.633333	0.4	0.5	0.333333	0.195180	0.769231
3	6	0.636364	0.5	0.5	0.500000	0.170941	0.846154
4	7	1.000000	1.0	1.0	1.000000	0.447214	1.000000
5	8	0.333333	0.0	0.0	0.000000	-0.154303	0.846154
6	9	0.076923	0.0	0.0	0.000000	-0.378412	0.714286
7	10	0.800000	0.5	1.0	0.333333	0.439155	0.846154

	Predicted_Teams_Set
0	{UTA, NYL, WAS, MIA}
1	{PHO, SAC}
2	{WAS, LAS}
3	{DET, CHA}
4	{MIN}
5	{NYL}
6	{ATL, NYL, HOU}
7	{SAC}

RESULTS VOTING-ENSEMBLE



Mean AUC – 0.544

Mean F1 – 0.362

RESULTS - TRAINING SEASON - TOP 5

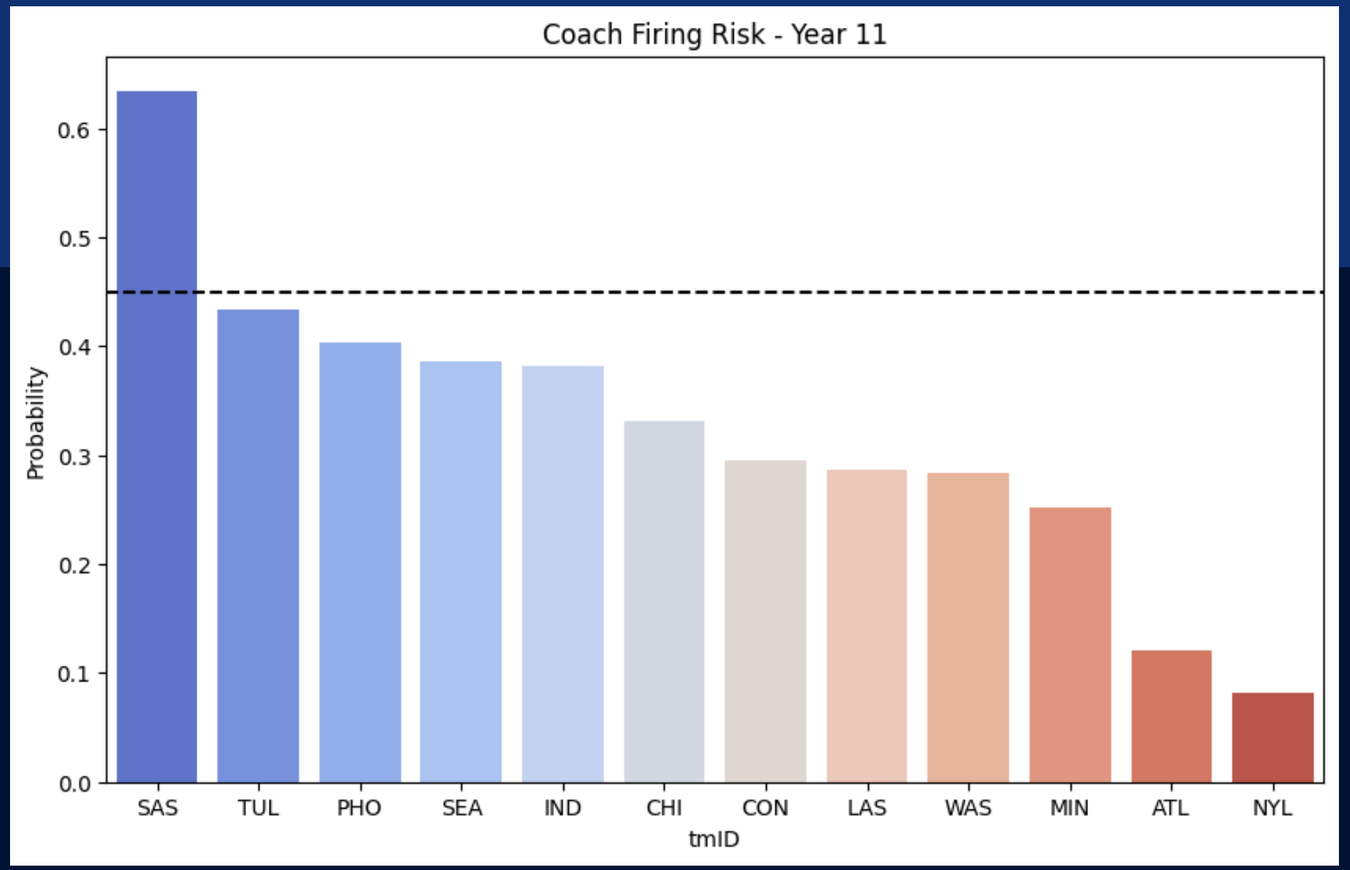
These are the model's most significant successful predictions:

- SAC (Year 4) - High Probability (83%)
- MIN (Year 7) - Medium Probability (52%)
- LAS (Year 5) - Above Threshold (48%)
- CHA (Year 6) - Above Threshold (46%)
- SAC (Year 10) - Above Threshold (46%)

RESULTS - TEST SEASON 11

The chart displays the predicted likelihood of a mid-season change for each team.

- SAS is the only team crossing our threshold (>0.45), flagging a predicted change.
- TUL, PHO, SEA and IND sit in a elevated risk zone, suggesting stability concerns despite not triggering a flag.



PREDICTING INDIVIDUAL AWARDS EVALUATION

Evaluation Objective

Predict individual award winners before the season starts, using only historical information from the previous year (T–1). For ROY, use pre-season biometric and context data.

Methodology — Walk-Forward Evaluation

For each year Y:

- Train the model using all seasons prior to Y (< Y).
- Evaluate performance on season Y.
- Repeat this process for years 3-10 (8 test years)

Note: Exception made for ROY, as rookies have no T-1 stats

How the Prediction Is Made

- Target: won_{award_name} (Binary: 1 = Won, 0 = Didn't Win).
- Model Architecture: RandomForestClassifier
  - n_estimators=200, max_depth=10, min_samples_split=5,
  - min_samples_leaf=2, class_weight='*balanced*',
  - random_state=42, n_jobs=-1
- Probabilities ranked

Evaluation Metrics

- Accuracy (Exact, Top-3 and Top-5)
- Average Winner Rank
- AUC - complementary especially to ordered accuracies

EVALUATION

We framed this problem as a binary classification task (won vs. didn't win) and tested multiple modeling approaches. While we did want to make Linear Regression work for this section, RandomForestClassifier proved itself to be solely a much better option due to the highly imbalanced nature of our data—where only 1 winner exists among hundreds of candidates per award each year.

We chose it for the following reasons:

- **Handles class imbalance better** - Built-in support for `class_weight='balanced'` penalizes errors on the rare winning class, improving predictions when winners are extremely underrepresented
- **Captures non-linear patterns** — Detects complex relationships like "high scoring AND high efficiency" or specific stat thresholds that Linear Regression cannot model
- **Consistent outperformance** — Across all test years, RandomForest significantly outperformed Logistic Regression and baseline models in both exact accuracy and top-3 ranking

CHOOSING AWARDS

Not every award was eligible to be chosen for prediction, we had to pick options that were eligible to use with just pre-season data, so we filtered the list of awards to predict to the following:

- Most Valuable Player (MVP)
- Defensive Player of the Year (DPOY)
- Most Improved Player (MIP)
- WNBA Finals Most Valuable Player (FMVP)
- Coach of the Year (COY)
- All-Star Game MVP (ASG_MVP)
- Rookie of the Year (ROY) - uniquely trained

This means we have chosen to leave out the following awards for the following reasons:

- Sixth Woman of the Year - 3 winners, 2 of which were rookies
- Sportsmanship Award - Purely subjective, no stats can be used
- Kim Perrot Sportsmanship Award - Purely subjective as well
- WNBA All-Decade Team / Honorable Mention - Retrospective career achievement award

Different features are used for different awards, so we'll go over them individually

MVP - FEATURES

The most straightforward and best performing model in our system.

Target variable: ‘won_most_valuable_player’

Features:

- **Performance based features**
  - points, rebounds, assists, steals, blocks, minutes and turnovers per game
  - field goals made/attempted (fg_pct), share of games started (gs_pct), games played and minutes
  - player_rating and efficiency (calculated from performance stats)
- **VS. League**
  - points, rebounds and assists per game versus league averages
  - player_rating and efficiency vs league average
  - 'is_elite_prev', to determine if player is an Elite player
- **Team Success**
  - team winning percentage, player’s team conference rank, if player’s team made playoffs, and team margin per game
- **Career stats**
  - years of experience, total awards and number of MVP awards

DPOY - FEATURES

In theory, it's also supposed to be one of the stronger models with robust defense features, however it turned out to be more subjective than we thought

Target variable: 'won_defensive_player_of_the_year'

Features:

- **Defensive Stats**
  - steals, blocks, defensive rebounds and rebounds per game
  - defensive player rating and efficiency
- **Overall Performance Context**
  - minutes per game, share of games started, player rating
- **Team Success**
  - team win percentage, team playoff appearances
- **Career History**
  - years of experience, total awards and total DPOY awards

MIP - FEATURES

This model struggled severely badly and had 0% accuracy even in Top-5 ranking. This is because this award is notably very **narrative** driven, and not statistics driven. Determining which player just so happens to be improving the most that year is highly subjective even if we assumed that they improved constantly over the previous years.

Target variable: ‘won_most_improved_player’

Features:

- **Improvement Metrics**
  - improvement of points, efficiency, player rating, and minutes per game
- **Career**
  - years of experience and total awards

FVMP - FEATURES

This model performed decently, although its accuracy did not improve as we looked deeper into Top 3 and Top 5 rankings of potential candidates, continuing at a steady 20%.

Target variable: ‘won_wnba_finals_most_valuable_player’

Features:

- **Playoff Stats**
  - post-season points, rebounds, assists, steals, blocks, games played, games started and minutes
- **Regular Season Performance**
  - points, rebounds and assists per game
  - player rating and efficiency
- **Team Playoff Success**
  - if team won championship last season, team playoff wins
- **Career**
  - years of experience, total awards and total WNBA Finals awards

ROY - FEATURES

Surprisingly, this model managed to predict one rookie, despite the fact that we could not use historical stats from the players, and only their biometric features, colleges, coach, team, etc

Target variable: ‘won_rookie_of_the_year’

Features:

- **Biometrics**
  - height and weight
- **Position**
  - uses boolean variables for different positions
- **College Reputation**
  - college score and if the player has a college value
- **Coach Track Record**
  - coach win percentage and playoff wins
- **Team Context**
  - team win percentage, team conference rank and team playoff appearances

COTY - FEATURES

This model ranked second-worst in performance across all predictions. Coaching success is inherently difficult to predict since a coach's impact depends heavily on roster quality, player health, and other factors beyond their control.

Target variable: ‘won_coach_of_the_year’

Features:

- **Team Performances**
  - team wins, win percentage, conference rank, playoff wins, team margin
  - if team made playoffs
- **Team Trajectory (T-1 vs T-2)**
  - win improvement and playoff appearance improvement
- **Career History**
  - career win percentage and playoff wins
  - years coaching
  - if they won COTY before

ASG - FEATURES

Because we couldn’t really tell who would be playing in the All-Star game through the test years’ rosters alone, we sort of had to also define features that would favor players who are more likely to be landing in the All-Star game in the first place, elites and such. Despite the obstacles, we still managed to predict one winner correctly.

Target variable: ‘won_all-star_game_most_valuable_player‘

Features:

- **Performance**
  - points, rebounds, assists, steals, blocks, minutes per game, field goal percentage
  - player rating and efficiency, share of games started, games played and total minutes
- **VS. League**
  - points, rebounds and assists per game vs league averages, player rating and efficiency vs league averages
- **Team Success Context**
  - win percentage, conference rank and if the team made playoffs
- **Career**
  - years of experience
  - total awards won, total ASG_MVP awards

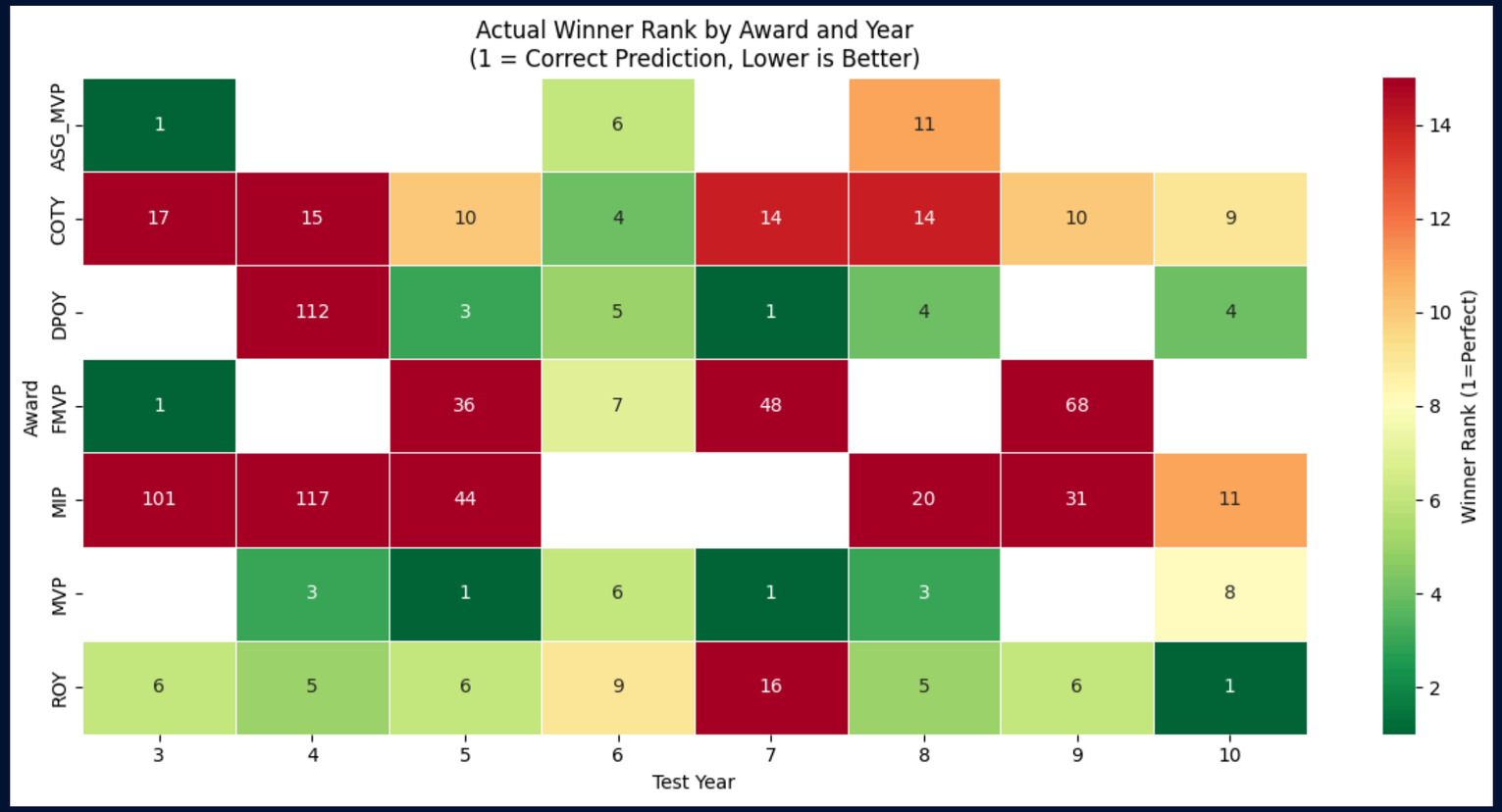
RESULTS - TRAIN DATA

Award	Years	Exact	Top-3	Top-5	Avg Rank	Avg AUC
MVP	6	2/6	67%	67%	3.7	0.979
DPOY	6	1/6	33%	67%	22.0	0.871
MIP	6	0/6	0%	0%	54.0	0.599
FMVP	5	1/5	20%	20%	32.0	0.563
ROY	8	1/8	12%	38%	6.8	0.843
COTY	8	0/8	0%	12%	11.6	0.294
ASG_MVP	3	1/3	33%	33%	6.0	0.643

These are the results obtained when testing years 3-10 on year 1 to Y-1 data for each year Y

MVP, DPOY and ROY return the best AUC values, judging by their ability to predict the winner is atleast somewhere closer to the top ranking, and showing the better ability to separate winner from non-winners

All the other ones, being below 0.7, although not too far below, show a poorer ability



Here, for every year, we can see the ranking of the actual winner in the prediction ranking, for when the award was actually handed out. The lower the number, the better, and if it's 1, it means the model predicted exactly who the winner would be.

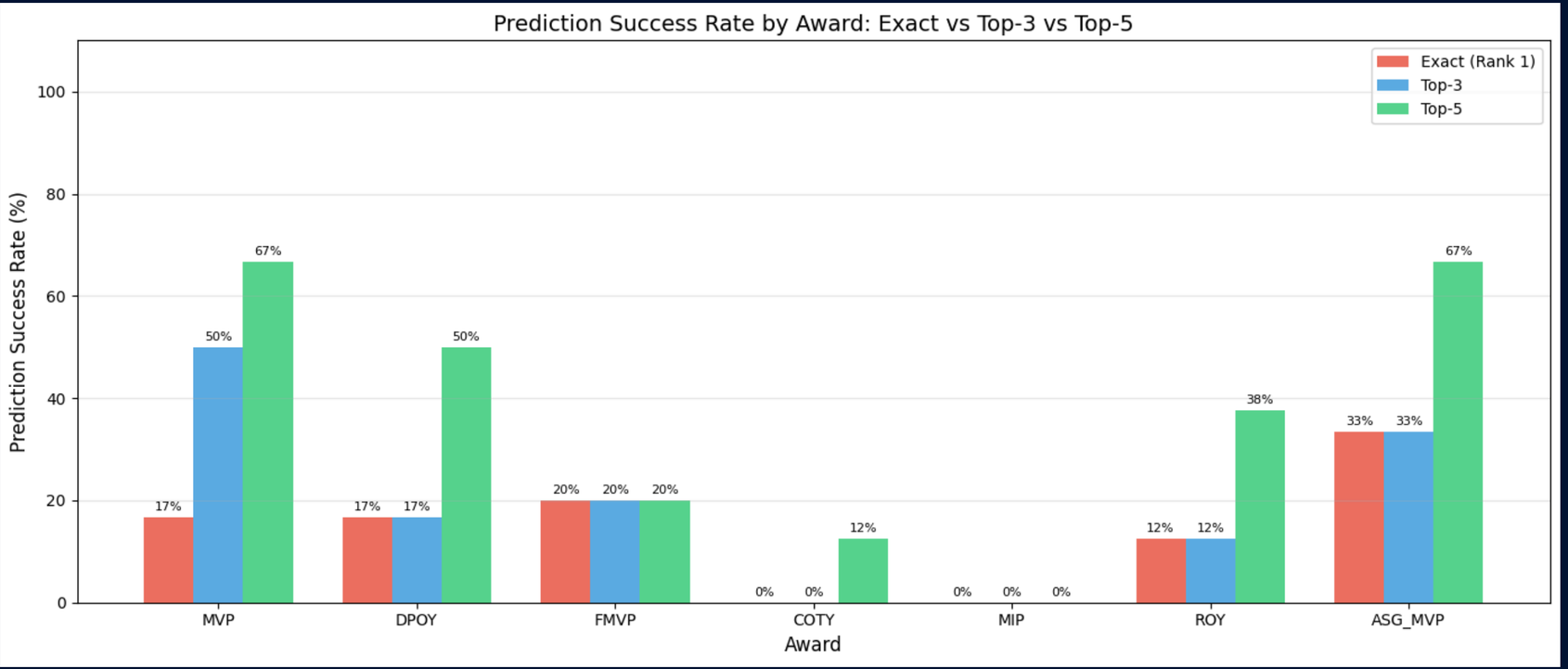
RESULTS - TRAIN DATA

Here we see that the MVP, DPOY, ROY, and ASG_MVP models, despite low exact prediction accuracy, perform better when considering Top 3 and Top 5 predictions, indicating that the true winner often ranks near the top.

FMVP's success rate does not improve when moving to Top 3 or Top 5, which aligns with the fact that the winner appears at rank #1 in some years but falls to #7 or lower in others.

COTY fails to predict the exact winner or place them in the Top 3, but shows some success within the Top 5.

MIP fails to predict the winner even within the Top 5.



This plot shows prediction success rates from years 3 to 10. It helps assess whether, even when the model misses the exact winner, it can still place the winner within the Top 3 or Top 5.

The Top 3 success rate is always equal to or higher than the exact prediction rate, and the Top 5 rate is always equal to or higher than the Top 3 rate, since Top 1 is included in Top 3 and Top 3 is included in Top 5.

RESULTS - TEST SEASON 11

Random Forest classifier trained on years 1-10 data and scored with predict_proba

Top 10 MVP Candidates:

playerID	tmID	mvp_proba	mvp_rank
tauradi01w	PHO	0.274470	1.0
jacksla01w	SEA	0.134837	2.0
powelni01w	NYL	0.084661	3.0
catchta01w	IND	0.059764	4.0
jonesas01w	CON	0.054545	5.0
parkeca01w	LAS	0.044935	6.0
youngso01w	SAS	0.029992	7.0
anosini01w	WAS	0.029890	8.0
dupreca01w	PHO	0.014819	9.0
pondeca01w	NYL	0.010000	10.0

Top 10 DPOY Candidates:

playerID	tmID	dpoy_proba	dpoy_rank
catchta01w	IND	0.314490	1.0
tauradi01w	PHO	0.274341	2.0
jacksla01w	SEA	0.259342	3.0
parkeca01w	LAS	0.104869	4.0
lyttlsa01w	ATL	0.099967	5.0
anosini01w	WAS	0.069931	6.0
mccouan01w	ATL	0.030000	7.0
beviltu01w	SAS	0.025000	8.0
perkiji01w	SAS	0.020000	9.0
cashsw01w	SEA	0.005000	10.0

Top 10 MIP Candidates:

playerID	tmID	mip_proba	mip_rank
parisco01w	ATL	0.280895	1.0
jacksti02w	TUL	0.180696	2.0
januabr01w	IND	0.160330	3.0
snowmi01w	CHI	0.105520	4.0
philler01w	IND	0.103842	5.0
montgre01w	CON	0.100237	6.0
holliqu01w	NYL	0.079321	7.0
ajavoma01w	WAS	0.069380	8.0
colemma01w	WAS	0.049414	9.0
walkede01w	CON	0.045409	10.0

Top 10 FMVP Candidates:

playerID	tmID	fmvp_proba	fmvp_rank
parkeca01w	LAS	0.187312	1.0
youngso01w	SAS	0.176013	2.0
douglka01w	IND	0.171345	3.0
tauradi01w	PHO	0.163124	4.0
catchta01w	IND	0.097899	5.0
willile01w	SEA	0.058230	6.0
pricear01w	ATL	0.055000	7.0
johnste01w	PHO	0.045000	8.0
pricear01w	ATL	0.045000	9.0
pondeca01w	NYL	0.044612	10.0

Top 10 ASG_MVP Candidates:

playerID	tmID	asg_mvp_proba	asg_mvp_rank
tauradi01w	PHO	0.143101	1.0
jacksla01w	SEA	0.078475	2.0
parkeca01w	LAS	0.064136	3.0
catchta01w	IND	0.049370	4.0
pondeca01w	NYL	0.028279	5.0
anosini01w	WAS	0.014775	6.0
braxtka01w	NYL	0.009604	7.0
bonnede01w	PHO	0.004910	8.0
lyttlsa01w	ATL	0.004866	9.0
desouer01w	ATL	0.004618	10.0

Top 10 ROY Candidates:

playerID	tmID	roy_proba	roy_rank
moorema01w	MIN	0.254748	1.0
colsosy01w	NYL	0.149090	2.0
robinda01w	SAS	0.135737	3.0
brelaje01w	NYL	0.129790	4.0
rileyan01w	TUL	0.128393	5.0
bjorkan01w	CHI	0.087388	6.0
hightal01w	CON	0.081106	7.0
montgal01w	NYL	0.078906	8.0
chestfe01w	NYL	0.064202	9.0
greenka01w	CON	0.047653	10.0

Top 10 COTY Candidates:

coachID	tmID	coty_proba	coty_rank
hugheda99w	SAS	0.189231	1.0
gaineco01w	PHO	0.178177	2.0
meadoma99w	ATL	0.126311	3.0
whisejo99w	NYL	0.095632	4.0
thibami99w	CON	0.087861	5.0
dunnli99wc	IND	0.080009	6.0
laceytr99w	WAS	0.069632	7.0
gilloje01w	LAS	0.050741	8.0
aglerbr99w	SEA	0.021761	9.0
richano99w	TUL	0.015306	10.0

CONCLUSIONS LIMITATIONS & FUTURE WORK

Conclusions

- Validated that historical data is a strong predictor for the upcoming season.
- Established a robust System that identifies risks and opportunities before the season begins.

Limitations

- As seen in EDA, historical data is skewed (some teams have 3x more games than others), and many events are statistically rare, making training difficult.
- The model cannot account for mid-season anomalies like sudden star player injuries, trades, or locker room conflicts.
- Chemistry and interpersonal relations are invisible to statistical datasets but are often the primary driver behind teams success.
- The model treats all games as equal, but in reality, some teams face significantly harder schedules or back-to-back fatigue scenarios that distort their win/loss record.

Future Work

- Shift from a static Pre-Season prediction to a dynamic model that updates weekly.
- Integrate external data (Social Media/News) to better capture the fan pressure on teams, players and coaches.