From Raw Plays to Chaos Features in College Football

Anisa Williams Braeden Mefford

College of Charleston

November 2025

Presentation Outline

What Are We Trying to Do?

- Build a **game-level feature set** to quantify “chaos” in college football.
- Starting from raw play-by-play, we want:
  - A clean play sequence for each game.
  - A reconstructed scoreboard by play.
  - Game-level features:
    - Lead change count.
    - Explosive play differential.
    - Win probability volatility.
- These feed into our overall **Chaos Factor**:

$$\text{Chaos} = w_1 \cdot \text{WinProbVolatility} + w_2 \cdot \text{LeadChanges} + w_3 \cdot \text{ExplosivePlayD}$$

Data Source: `cfbfastR`

- We use the `cfbfastR` package to pull NCAA football play-by-play data.
- Seasons: **2018–2023**.
- Dataset type: **team-independent play-by-play**.
- Each row = one play, with:
  - `game_id`, `pos_team`, `def_pos_team`.
  - `pos_team_score`, `def_pos_team_score`.
  - `home_wp_after`: home win probability after the play.

Downloading Play-by-Play with `cfbfastR`

- Load required packages:
  - `cfbfastR`, `dplyr`, `tidyr`, `janitor`, `purrr`.
- Choose seasons:

```
seasons <- 2018:2023
```

- Pull play-by-play:

```
pbp <- load_cfb_pbp(seasons = seasons)
```

- Result: a large data frame with all plays from 2018–2023 Power 5 games (after filtering).

Raw Play Structure

- For each play we observe:
  - `game_id`: which game.
  - `pos_team`, `def_pos_team`: offense and defense.
  - `pos_team_score`, `def_pos_team_score`: cumulative scores.
  - `home_wp_after`: win probability for the home team.
- Issues preventing direct feature computation:
  - Play order is not consistently encoded across seasons.
  - Scores are stored relative to **possession**, not per-team columns.

Problem 1: Inconsistent Play Identifiers

- Different columns used to index plays:
  - `game_play_number` in some seasons.
  - `id_play` in others.
  - Sometimes neither is ideal.
- But we must know the **exact sequence of plays**:
  - 1st play, 2nd play, . . . , last play for each `game_id`.

Solution: A Robust `play_order` Variable

- We define a helper function (in R) to select an order column:
  - Use `game_play_number` if present.
  - Else, use `id_play` with `rank(..., ties.method = "first")`.
  - Else, fall back on row number within each game.
- Then compute:

  ```
  play_order = 1, 2, 3, ...
  ```

  within each `game_id`.
- Now every game has a clean chronological progression.

Problem 2: Scores Are Possession-Relative

- Raw columns:
  - `pos_team_score`: score of the offense.
  - `def_pos_team_score`: score of the defense.
- No direct “score for Team X and Team Y at this play” layout.
- For chaos features, we want a scoreboard-like structure:

$$(\text{game\_id}, \text{play\_order}, \text{Team 1 Score}, \text{Team 2 Score}).$$

Strategy: Long-to-Wide Scoreboard

- For each play, we:
  - Create two rows in long format:

$$(\text{game\_id}, \text{play\_order}, \text{team}, \text{score}).$$

    where `team` is either `pos_team` or `def_pos_team`.
- Then we:
  - Sort by `game_id`, `play_order`.
  - Forward-fill `score` within each (game, team).
  - Drop duplicates so each (game, play, team) appears once.
  - Pivot to wide format with one column per team's score.
- Output: a **scoreboard** table with scores for both teams on every play.

Aligning with Home and Away Teams

- We join in game metadata:
  - `home`, `away` team names per `game_id`.
- If the scoreboard columns match the home and away labels:
  - We can take:

$$\text{margin} = \text{score\_home} - \text{score\_away}.$$

- If not, we:
  - Identify the two main team columns by frequency.
  - Use the first two numeric score columns to approximate the margin (for sign).
- This prepares us to compute **lead changes** game by game.

Defining Lead Changes

- For each play, define the score margin:

$$\text{margin}_t = \text{HomeScore}_t - \text{AwayScore}_t.$$

- Convert margin into a discrete **lead sign**:

$$\text{lead\_sign}_t = \begin{cases} +1 & \text{if margin}_t > 0 \\ -1 & \text{if margin}_t < 0 \\ 0 & \text{if margin}_t = 0 \end{cases}$$

- A **lead change** occurs when:

$$\text{lead\_sign}_t \neq \text{lead\_sign}_{t-1},$$

ignoring transitions where either sign is 0 (ties).

Lead Change Count per Game

- For each `game_id`, we:
  - Sort plays by `play_order`.
  - Compute `lead_sign` at each play.
  - Count the number of valid sign flips.
- This gives a game-level feature:

$$\text{LeadChangeCount}_{\text{game}} = \sum_t \mathbf{1}\{\text{lead\_sign}_t \neq \text{lead\_sign}_{t-1}\}.$$

- Interpretation:
  - High values: back-and-forth contests.
  - Low values: one-sided games with a stable leader.

Defining Explosive Plays

- Using standard football thresholds, we define an **explosive play** as:
  - A rush gaining at least 12 yards, or
  - A pass gaining at least 16 yards.
- For each play, we create an indicator:

$$\text{explosive} = \mathbf{1}\{\text{rush} = 1 \wedge \text{yards\_gained} \geq 12\} + \mathbf{1}\{\text{pass} = 1 \wedge \text{yards\_gai}$$

- This is computed in the cleaned play-by-play data.

Explosive Plays by Team and Game

- For each game, we aggregate by offense:

$$\text{ExplosiveOffense}_{\text{game,team}} = \sum_{\text{plays by team}} \text{explosive}.$$

- Then we compress to a single game-level statistic:

$$\text{ExplosivePlayDelta}_{\text{game}} = \max_{\text{teams}} \text{ExplosiveOffense} - \min_{\text{teams}} \text{ExplosiveOffe}$$

- Interpretation:
  - Large delta: one team generated many more explosives.
  - Small delta: explosive plays were balanced.

Win Probability at Each Play

- `cfbfastR` provides:

$$\texttt{home\_wp\_after} \in [0, 1],$$

  the home team's win probability <u>after</u> each play.
- Once we have a consistent play order, each game yields a sequence:

$$\{\texttt{home\_wp\_after}_1, \texttt{home\_wp\_after}_2, \ldots, \texttt{home\_wp\_after}_T\}.$$

- This sequence captures how beliefs about the home team's chances evolve over time.

Defining Win Probability Volatility

- We define win probability volatility for each game as:

$$\text{WinProbVolatility}_{\text{game}} = \mathrm{sd}\left(\{\texttt{home\_wp\_after}_t\}_{t=1}^{T}\right),$$

where sd is the sample standard deviation.

- Computed in R via:

```
pbp_clean %>% group_by(game_id) %>% summarise(win_prob
```

- Interpretation:
  - High volatility = chaotic, swingy game.
  - Low volatility = stable, predictable game.

Combining All Features

- From our R pipeline, we end up with:
  - **Lead changes**: $\text{LeadChangeCount}_{\text{game}}$
  - **Explosive play differential**: $\text{ExplosivePlayDelta}_{\text{game}}$
  - **Win probability volatility**: $\text{WinProbVolatility}_{\text{game}}$
- We join these back to basic game info:
  - `game_id`, `home`, `away`.
- Final object: `game_features`, one row per game:

$$(\texttt{game\_id}, \texttt{home}, \texttt{away}, \text{LeadChangeCount}, \text{ExplosivePlayDelta}, \text{WinF}$$

How These Feed the Chaos Factor

- Our Chaos Factor is constructed as:

  $$\text{Chaos} = w_1 \cdot \text{WinProbVolatility} + w_2 \cdot \text{LeadChangeCount} + w_3 \cdot \text{Explosive}$$

- All three components come from the data pipeline we just walked through.
- Key benefits:
  - Uses only play-by-play and win probability data.
  - Is reproducible and extensible to future seasons.
  - Captures multiple dimensions of “game chaos”.

Pipeline Summary

- **Step 1: Collect** play-by-play data with `cfbfastR` (2018–2023).
- **Step 2: Standardize play order** per game.
- **Step 3: Reconstruct** a per-play scoreboard for both teams.
- **Step 4: Compute features**:
  - Lead change count.
  - Explosive play differential.
  - Win probability volatility.
- **Step 5: Merge** into a game-level table that feeds the Chaos Factor.

Why This Matters

- These features quantify:
  - How often control of the game changes (lead changes).
  - How uneven big plays are (explosive play delta).
  - How uncertain the outcome feels over time (win prob volatility).
- Together, they give a richer picture of game dynamics than Elo alone.
- This data pipeline is the foundation for:
  - Chaos-based rankings,
  - Upset prediction,
  - And deeper storytelling about “crazy” games.