MEASURING AND PREDICTING TILT AFTER LARGE LOSSES IN ONLINE POKER CASH GAMES

Max Héberlé
HEC Lausanne
MSc. In Management ; max.heberle@unil.ch

Abstract— This project leverages anonymized online cash-game hand histories to study *tilt*, i.e., decision-making changes following a salient loss. We build an end-to-end pipeline from hand-history parsing to the construction of behavioral indicators by player and stake level. An exploratory analysis characterizes playing profiles and compares behavior before and after the event (tendencies to fold and betting dynamics). Finally, we train machine learning models to predict the magnitude of these changes from pre-event player profiles, using a cross-validation design that prevents information leakage across players. The goal is both descriptive (to characterize tilt patterns) and predictive (to anticipate which profiles are most likely to change after a major loss).

Index Terms— Online poker, cash game, tilt, player behavior, exploratory data analysis, machine learning.

I. Introduction

Cash game poker is an environment in which decisions follow one another and uncertainty is constant. In this context, players commonly use the term tilt to describe a period during which the way they play changes after a negative event, in particular a significant loss. Even though this notion is very present in practice, studying it quantitatively raises several difficulties: playing styles are highly heterogeneous, behaviors can evolve depending on stake levels (stakes), and the data can vary according to very abstract criteria. This environment is interesting because it combines repeated decision-making, risk-taking, and feedback linked to results, all of which can generate short-term behavioral adjustments.

In this project, we use a dataset from the Zenodo repository A Dataset of Poker Hand Histories (Kim, 2024). The goal is not to comment on hands individually, but to transform raw traces into a set of indicators that are comparable across players and across stakes. This step is central: it requires parsing, ensuring consistency of identifiers, and normalizing amounts in order to make analyses meaningful across different blind levels.
The overall objective of the project is to characterize and measure potential behavioral changes after a salient loss, defined by a rule that we will discuss later. First, we carry out an exploratory analysis to describe the dataset, understand the structure of the observations, and examine the variability of player profiles across stakes. Next, we set up a before/after study framework around the events in order to assess whether certain behavioral tendencies change systematically. Finally, we will end with a predictive approach aimed at determining whether the magnitude of a change can be anticipated from information observed before the event, using Machine Learning. It is important to note that the objective is not to draw causal conclusions, since the data are observational and many unobserved factors may influence decisions; rather, we seek to propose a coherent measurement framework and to assess whether pre-event information contains a signal about post-event evolution.

II. Research question and relevant literature

This project focuses on tilt in online cash-game poker, understood in the practical sense as a change in the way one plays after a salient negative outcome, in particular a significant loss. Using observational data collected via scraping and then prepared for analysis, the objective is to build a reproducible framework that identifies loss events according to an operational rule, compares behavior before and after these events, and examines whether the information available before the event contains a signal that makes it possible to anticipate the subsequent evolution. The challenge is as much methodological as empirical: it is about moving from a concept used by players to measures that are comparable across individuals and across game contexts, without claiming a causal interpretation.

This question relates to a broader literature on the influence of past outcomes on decision-making under risk. Prospect Theory highlights

preferences that depend on a reference point and an asymmetry between gains and losses (loss aversion), which provides a general framework for understanding why the same person may react differently after a gain or after a loss, depending on what becomes the psychological reference point (Kahneman and Tversky, 1979). In the same spirit, Thaler and Johnson (1990) document two often-discussed behavioral regularities: the house-money effect (increased risk-taking after gains) and the break-even effect (a tendency to take more risks after losses in order to "break even") (Thaler and Johnson, 1990). These results motivate the idea that a negative event can modify, over a short horizon, subsequent decisions, even if the strategic environment remains broadly comparable.

Another important point of contact is the literature on loss-chasing in gambling behaviors, which describes the tendency to continue and/or intensify gambling after losses in order to compensate for them. Recent work emphasizes that loss-chasing covers several expressions (for example within a session or between sessions) and that empirical conclusions depend strongly on how the concept is operationalized (Zhang & Clark 2020). This remark is directly relevant here: studying tilt quantitatively requires explicitly defining what constitutes an event, as well as the choice of "before/after" comparison windows.

Finally, the poker-specific literature has proposed tools dedicated to tilt, notably the Online Poker Tilt Scale (OPTS), developed and validated to measure the frequency of tilt episodes among online players (Moreau et al., 2017). These contributions are mainly based on self-reported data. The present project fits in as a complement, relying on behavioral traces from hand histories, with an emphasis on data processing, comparability across stakes, and the evaluation of prediction in a framework that limits information leakage across players.

III. Methodology

We begin by transforming the raw data into usable structured tables. The project relies on a clear separation between information at the hand level (identifiers, stake, timestamp, stack variables, and outcome variables) and information at the action level (action sequence and action type of the studied player). This phase includes parsing, ensuring consistency of identifiers, and creating a chronological ordering of hands by player and by stake. Monetary amounts are then normalized into Big Blinds (BB) to make indicators comparable across stakes.

From the structured hands, we construct an operational rule to identify loss events. An event is triggered when a sequence of 1 to 3 consecutive lost hands simultaneously satisfies constraints on the relative loss measured as a loss of at least 20% of the stack at the start of the hand (net_pct ≤ −0.20), a loss of at least 10 BB (net_bb ≤ −10), and a stack depth at the beginning of the episode of at least 40 BB (start_bb ≥ 40). When several sequence lengths satisfy these conditions, we retain the shortest one (event_len ∈ {1,2,3}) in order to define the most "local" starting point possible. Finally, a cooldown mechanism of 20 hands is applied to avoid counting multiple events that are too close in time for the same player and the same stake.

Once events are defined, we set up a before/after comparison. For each event, we extract two fixed-length time windows: a PRE window including up to 20 hands before the event, and a POST window including up to 20 hands after the end of the event. If the full window is not available on one side, we keep the available information over as many hands as possible. On these windows, we compute behavioral indicators centered on the studied player, distinguishing two families: action frequencies (fold, call_or_check, bet_or_raise) and sizing metrics (bet/raise amounts, normalized in BB), summarized by simple statistics (mean, median). The effect of an event is then summarized by a POST−PRE difference for each metric.

For descriptive inference, the analysis is conducted stake-by-stake in order to account for heterogeneity across stake levels. For each stake and each metric, we report aggregated statistics of the POST−PRE differences. The significance of a systematic change is assessed using a paired nonparametric test (Wilcoxon Signed-Rank Test), applied to the event-by-event differences. In addition, confidence intervals are obtained via bootstrap on the event-level differences, with a fixed number of resamples, in order to provide an estimate of uncertainty around mean effects.

Finally, we formulate the predictive part as a supervised regression problem: the objective is to predict the magnitude of a post-event change from a set of variables measured before the event. This step is performed stake-by-stake, and the evaluation is organized with 5-fold GroupKFold cross-validation, grouped by player_id, to avoid any information leakage between observations from the same player. Three models are compared: Ridge Regression, k-Nearest Neighbors (kNN), and

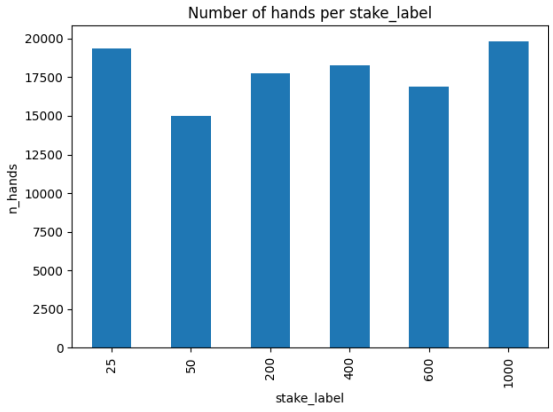
AdaBoost. Performance is evaluated on Out-of-Fold (OOF) predictions using Mean Squared Error (MSE), Mean Absolute Error (MAE), and R-squared (R^2), and compared to a naive baseline defined in each fold as the mean of the target on the fold's training sample. This combination (before/after analysis + out-of-sample prediction with player-level separation) makes it possible to address the research question without assuming a causal relationship and while controlling for obvious biases related to reusing the same player in train and test.

IV. Dataset

The dataset used in this project comes from the Zenodo repository entitled A Dataset of Poker Hand Histories (Kim, 2024). The source description states that the corpus contains anonymized hand histories from scraped logs over a given period, including PokerStars hands at different betting levels.

stake	hands	players	tables	actions
25	19331	249	607	239542
50	15013	235	352	145999
200	17755	240	448	169225
400	18261	239	403	171254
600	16885	243	355	162075
1000	19838	236	412	163948

Dataset overview by stake (hands, players, tables,actions)



Number of hands per stake level in the analysis dataset.

The cleaning was designed as a rerunnable pipeline, with clear intermediate outputs, in order to move from raw text files to two structured tables (hands and actions) usable for EDA and Machine Learning.

1. Conversion of raw files to parquet (hands_parquet and player_hands_parquet)
   Hand histories in text format are traversed file by file and split into sections corresponding to hands. At this stage, we filter hands by keeping only those whose venue field corresponds to PokerStars. For each retained hand, we extract the available metadata (timestamp, table, small blind, big blind) as well as the players, stacks, seats, and actions lists. This information is stored in a hands table (one row per hand), keeping certain columns in the form of serialized lists (JSON) in order to retain all content without losing information. In parallel, we build a player_hands table (one row per player and per hand) but only for a subset of players, selected deterministically via hashing (about 20% of players). A hand is kept as soon as at least one selected player appears in it, which reduces the volume while remaining reproducible. Outputs are written to parquet in batches (flush) and organized by partitions, in particular by stake_label and (for player_hands) by player_bucket.
2. Reconstruction of a per-player timeline (timeline_parquet)
   From player_hands_parquet, a DuckDB step reconstructs a stable chronological ordering of hands for each player at each table and stake, sorting by timestamp and then by hand identifier, and assigning an increasing rank (row number). This timeline is exported in parquet, partitioned by stake_label and player_bucket, which then makes it possible to quickly retrieve a player's hands without loading the entire corpus.
3. Balanced sampling of players and export to text transcripts
   To limit the cost of subsequent steps while keeping a comparable base across stakes, we draw a deterministic sample of players per stake (via an order based on md5(player_id)), then we export for each player a text transcript containing a fixed number of hands in chronological order. These exports are protected against duplicates (a "safe" write via a temporary file then rename) and a manifest is generated to trace what was actually produced.
4. Final structuring of transcripts into two tables (hands and actions)
   The transcripts are then reparsed with

regular expressions: we identify each “hand” block, extract the header (timestamp, hand identifier, table, stack information and net result), then split the sequence of actions by street. Each hand feeds one row in a structured hands table, while each action feeds an actions table (one row per action) with the street, the action type and, when present, the amount (notably for bet/raise). Writing is done in parquet chunks and execution is resumed automatically thanks to a manifest/skip system, which makes the whole process idempotent (rerunning the notebook does not recreate duplicate files).

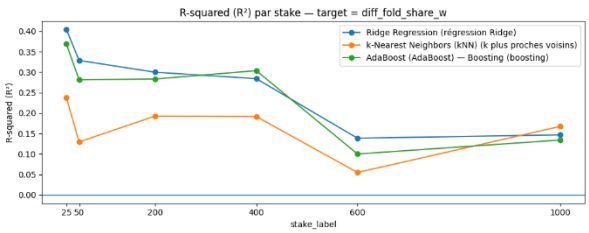
V. Implementation of the code

The implementation is structured into three distinct notebooks (data cleaning, EDA, Machine Learning) in order to separate costly steps (parsing and restructuring) from iterative steps (analyses and models). In each notebook, a configuration cell centralizes input/output paths (Google Drive) as well as rerun switches (for example FORCE_REBUILD_*). The code checks whether outputs exist before recomputing (dir_has_files, count_parquet, count_txt), which makes it possible to rerun the pipeline without overwriting or duplicating files.

The data cleaning notebook emphasizes robustness and scalability. Parsing is encapsulated in dedicated functions (split_hand_blocks, parse_hand_block, split_actions_to_streets, parse_action_line, parse_action). To limit the memory footprint, parquet writes are done in batches via buffers and flush functions (flush_hands, flush_ph, flush_all), and outputs are partitioned by stake_label (and, if necessary, by a deterministically computed player_bucket). The reproducible selection of a subset of players relies on hashing functions (md5_int, keep_player, player_bucket), making the sample independent of the reading order. Per-player text exports are driven by export_player_transcript and tracked in exports_manifest.csv (ok / skipped / error). The final restructuring uses a restart mechanism via _manifest_processed.txt, and file writing is secured by a “safe” write (temporary file then os.replace) to avoid corrupted outputs in case of interruption.

In the EDA notebook, the implementation loads the hands and actions tables and then performs simple checks (missing values, timestamps, types). The analysis focuses on the actions of the studied player (is_hero == True) and groups actions into three categories via a mapping (fold, call_or_check, bet_or_raise). Stake-by-stake descriptive tables (counts/proportions, distributions by street) document variability and check coverage. For the before/after study, an events table is built from hands: sorting by player and stake (seq_no), detection of consecutive-loss episodes, then application of a cooldown to avoid events that are too close. PRE and POST windows are defined via an explicit join (build_window: event × delta keys then join on a lookup table), and coverage checks measure the share of events with at least one decision action. Metrics are computed via groupby aggregations, while bet/raise sizes are handled separately (prep_sizes: bet_or_raise filtering, numeric conversion, cleaning, conversion to BB, mean/median aggregation). POST−PRE differences are then synthesized stake-by-stake with Wilcoxon (wilcoxon_p) and bootstrap confidence intervals (bootstrap_ci_mean) using a deterministic seed (stake/metric). The tables needed for Machine Learning are exported to _exports (CSV and, when possible, parquet).

In the Machine Learning notebook, the implementation starts from the events dataset exported by the EDA (ml_events) and is organized stake-by-stake. The number of folds is handled safely (n_splits = min(5, n_players); stake skipped if GroupKFold is not possible). Explanatory variables are defined in FEATURES and the modeling is supervised regression on continuous targets. Three scikit-learn pipelines are compared: Ridge (median imputation, standardization, alpha=1.0), kNN (median imputation, standardization, n_neighbors=15), and AdaBoost (median imputation, AdaBoostRegressor with DecisionTreeRegressor max_depth=2, min_samples_leaf=5, n_estimators=400, learning_rate=0.05, loss="linear"). Evaluation uses GroupKFold grouped by player_id and cross_validate (neg_mean_squared_error, neg_mean_absolute_error, r2) with sign conversion to report MSE, MAE, and R².



Out-of-fold R-squared (R²) by stake for each model

The notebook also generates Out-of-Fold (OOF) predictions for diagnostics (y_true vs y_pred, residuals) and compares each model to an internal per-fold baseline (mean of the target

on the training set). For Ridge, coefficients are extracted per fold and then aggregated. All outputs (metrics, OOF, tables, and figures) are saved in _exports for direct reuse in the report.

VI. Results

EDA results: average changes after a big loss

Table 1 summarizes event coverage after filtering and applying the cooldown: number of events and unique players by stake (stake_label).

stake	events	players
25	253	150
50	167	114
200	172	120
400	195	111
600	148	98
1000	206	117

Table 1 — Event coverage by stake (after cooldown).

Overall, the coverage is relatively homogeneous (about 148 to 253 events depending on the stakes), which allows stake-by-stake comparison, while keeping in mind that some sub-analyses (e.g., sizing) use fewer events due to missing values.

1.1 Fold share and aggressiveness (Bet/Raise Share)

Table 2 and Figures 1–2 present the mean variation of two behavioral metrics: fold share (Δ Fold) and the share of bets/raises among non-fold decisions (Δ Bet/Raise). The 95% confidence intervals (95% CIs) are bootstrap intervals for the mean Δ (POST−PRE). By contrast, p-values are computed using the paired Wilcoxon Signed-Rank test on event-level Δ values, which tests whether the median Δ differs from 0.

Stake	N_events	Δ Fold	p(Fold)	Δ Bet/Raise	p(Bet/Raise
25	234	-2.88 pp [-5.41, -0.51]	0.1111	-1.24 pp [-3.30, +0.79]	0.3137
50	156	+0.36 pp [-3.52, +4.08]	0.9849	-0.31 pp [-2.75, +2.15]	0.9870
200	159	-4.07 pp [-7.24, -0.86]	0.0091	+0.43 pp [-1.89, +2.59]	0.4837
400	182	-1.89 pp [-4.99, +1.24]	0.1587	-0.88 pp [-3.33, +1.54]	0.4321
600	133	-1.04 pp [-4.62, +2.29]	0.6562	-0.96 pp [-3.71, +1.77]	0.4148
1000	192	-4.65 pp [-7.34, -1.84]	0.0009	+1.59 pp [-0.47, +3.76]	0.2122

Table 2 — POST - PRE variations in fold and Bet/Raise Share (95% bootstrap CI, p-value

Main result (fold share). Two stakes stand out with a statistically significant decrease in fold share: stake 200 (Δ Fold = -4.07 percentage points, p = 0.0091) and stake 1000 (Δ Fold = -4.65 pp, p = 0.0009). For the other stakes, mean Δ estimates remain close to 0 and the bootstrap 95% CIs include 0; likewise, the Wilcoxon Signed-Rank test does not detect a systematic shift in the distribution of Δ (p > 0.05). This suggests that at these stakes, either the mean effect is truly close to zero, or there is strong inter-player heterogeneity (some increase, others decrease), which dilutes the aggregated effect and reduces statistical power.

Aggressiveness (Bet/Raise Share). No stake shows a robust change after the event: the 95% CIs include 0 and p-values remain well above usual thresholds, suggesting that any mean effect is either null or too small relative to inter-player variability and the noise inherent to this metric (aggressiveness depends heavily on a few decisions and table context, and opposite reactions can offset each other on average). A slight increase is nevertheless observed at stake 1000 (Δ Bet/Raise = +1.59 pp), but it remains non-significant (p = 0.2122), indicating that with the available data we cannot clearly distinguish this trend from a simple sampling effect.

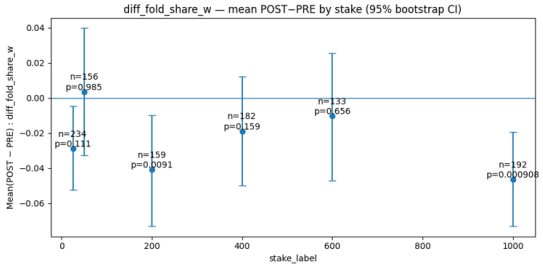


Figure 1 — Δ fold share (POST - PRE) by stake: mean and 95% bootstrap CI (annotation: n, p-value).

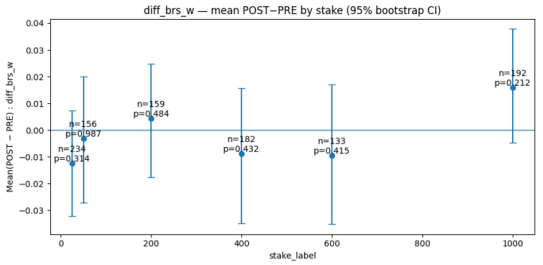


Figure 2 — Δ Bet/Raise Share (POST - PRE) by stake: mean and 95% bootstrap CI (annotation: n, p-value).

1.2 Bet/raise sizing (mean and median, in BB)

Figures 3–4 and Table 3 focus on bet/raise sizing, expressed in big blinds (BB). Two aggregates are reported: the change in mean size (Δ Mean size) and median size (Δ Median size) in a 20-hand window after the event, compared with a 20-hand window before.

Stake	N_sizing	Δ Mean size	p(Mean size)	Δ Median size	p(Median size)
25	207	+0.16 BB [-0.79, +1.03]	0.2009	+0.58 BB [-0.02, +1.22]	0.0476
50	126	+0.55 BB [-0.36, +1.54]	0.1631	+0.11 BB [-0.68, +0.84]	0.1605
200	141	+0.37 BB [-1.06, +1.63]	0.0982	-0.42 BB [-1.46, +0.45]	0.4993
400	157	+0.42 BB [-0.57, +1.42]	0.7819	+0.05 BB [-0.59, +0.73]	0.9620
600	122	-0.11 BB [-1.11, +0.90]	0.8161	-0.02 BB [-0.80, +0.78]	0.6636
1000	174	+0.19 BB [-1.08, +1.40]	0.5403	+0.01 BB [-0.90, +0.76]	0.2965

Table 3 — POST - PRE sizing variations (95% bootstrap CI, p-value).

Main result (median). At stake 25, we observe a statistically significant increase in the median size of bets/raises (Δ Median size = +0.58 BB, p = 0.0476), with a 95% CI very close to 0 on the lower side, indicating a small but detectable effect. This “borderline” significance can be explained by the fact that the median is a robust statistic, little influenced by a few very extreme

bets: if the change affects the “core” of sizings (a slight frequent increase), the median captures it better than the mean. For the other stakes, no clear variation is detected: the 95% CIs include 0 and p-values are non-significant, which is consistent with a small mean effect and/or inter-player heterogeneity (some increase, others decrease), which dilutes the aggregated signal.

Mean size variations show no robust evidence of a systematic change across stakes: bootstrap 95% CIs for the mean Δ include 0, and Wilcoxon p-values are all > 0.09. The relatively wide 95% CIs reflect substantial variability in bet/raise sizes and the presence of extreme observations (heavy tails), which is typical of poker sizing distributions. In practice, a few large post-event sizings (or, conversely, a few very small ones) can shift the mean without reflecting a systematic change in behavior: the mean is therefore more sensitive to noise and extreme values, which reduces statistical power and explains the lack of significance despite some visible trends.

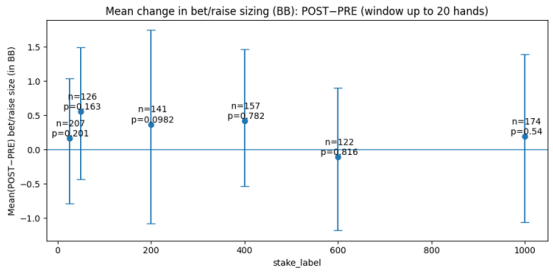


Figure 3 — Δ Mean bet/raise size (POST - PRE) by stake: mean and 95% CI (window up to 20 hands).

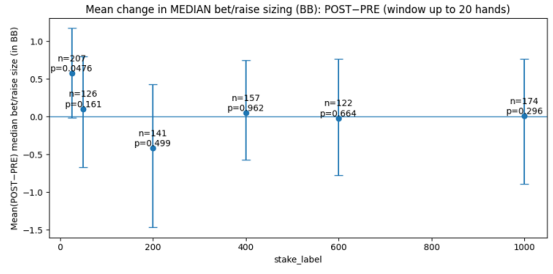


Figure 4 — Δ Median bet/raise size (POST - PRE) by stake: median and 95% CI (window up to 20 hands).

2 Machine Learning results: predicting the POST – PRE variation

The objective of the modeling is to predict, from a player’s characteristics before the event (PRE window), the behavior variation observed after the event (POST - PRE), by training separate models for each stake. Two continuous targets are studied: diff_fold_share_w and diff_median_size_bb.

Validation relies on GroupKFold (5 folds) with group = player_id, so that all observations of a player are confined to a single fold. The predictions analyzed below are Out-of-Fold (OOF), i.e., produced on observations not seen during training. Performance is reported via Mean Squared Error (MSE), Mean Absolute Error (MAE) and R-squared (R^2), with comparison to a naive baseline (predicting the mean).

2.1 Comparison of the 3 models and performance by stake

Table 4 summarizes, for each target and each stake, the best model selected according to the Mean Squared Error (MSE), as well as the R-squared (R^2) obtained and the Mean Squared Error (MSE) gain relative to the baseline.

Target	Stake	Best model (by MSE)	R-squared (R^2)	MSE gain vs baseline (%)
diff_fold_share_w	25	Ridge Regression (Ridge regression)	0.404	41.23
diff_fold_share_w	50	Ridge Regression (Ridge regression)	0.328	34.21
diff_fold_share_w	200	Ridge Regression (Ridge regression)	0.300	30.03
diff_fold_share_w	400	AdaBoost (AdaBoost) — Boosting (boosting)	0.304	31.48
diff_fold_share_w	600	Ridge Regression (Ridge regression)	0.138	15.65
diff_fold_share_w	1000	k-Nearest Neighbors (kNN)	0.167	17.80
diff_median_size_bb	25	Ridge Regression (Ridge regression)	0.335	34.46
diff_median_size_bb	50	Ridge Regression (Ridge regression)	0.573	57.44
diff_median_size_bb	200	Ridge Regression (Ridge regression)	0.800	80.13
diff_median_size_bb	400	Ridge Regression (Ridge regression)	0.388	39.10
diff_median_size_bb	600	Ridge Regression (Ridge regression)	0.260	27.88
diff_median_size_bb	1000	Ridge Regression (Ridge regression)	0.618	62.53

Table 4 — Best model by stake and by target (selection by Mean Squared Error (MSE)).

For the target diff_median_size_bb, Ridge Regression (Ridge regression) dominates systematically and performance is markedly higher, with R-squared (R^2) ranging from 0.26 to 0.80. The best predictability level is observed at stake 200 (R^2 = 0.80).

For the target diff_fold_share_w, performance is more modest and more variable across stakes (R^2 between 0.14 and 0.40), suggesting that fold share variation after a big loss is less explained by the selected PRE features. Ridge Regression (Ridge regression) is most often best, but AdaBoost (AdaBoost) — Boosting (boosting) is selected at stake 400 and k-Nearest Neighbors (kNN) at stake 1000 under the Mean Squared Error (MSE) criterion.

2.2 Interpretability: Ridge coefficients (standardized features)

Figures 5–6 present the mean coefficients of Ridge Regression (Ridge regression) (average over folds). With features standardized, coefficients are comparable across variables: a positive coefficient indicates that a higher PRE value is associated with a higher POST - PRE variation (for a given stake), all else being equal.

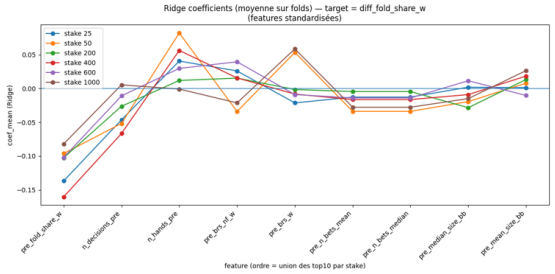


Figure 5 — Ridge Regression (Ridge regression): mean coefficients (target diff_fold_share_w, standardized features).

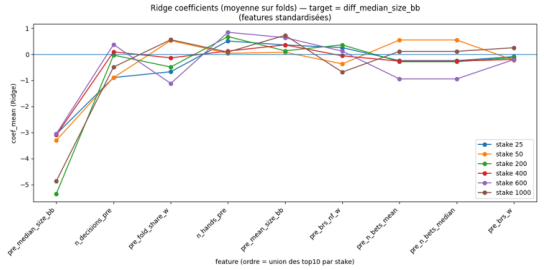


Figure 6 — Ridge Regression (Ridge regression): mean coefficients (target diff_median_size_bb, standardized features).

Key points (qualitative reading). On diff_median_size_bb, the coefficient associated with pre_median_size_bb is strongly negative on all stakes, which corresponds to a “regression to the mean” effect: players who were already betting larger in median before the event are predicted to reduce their sizing more after the event (more negative variation).

On diff_fold_share_w, pre_fold_share_w also shows a marked negative coefficient on all stakes: a player already very “foldy” before the event is predicted to have a larger decrease in fold share (or a smaller increase) after the event, which is consistent with a dynamic of returning toward more average behavior.

2.3 Out-of-Fold (OOF) diagnostics: y_true vs y_pred and residuals

Figures 7–18 show, for each of the three models, (i) the y_true vs y_pred (OOF) scatter plot and (ii) the residual histogram (y_true - y_pred). The diagonal corresponds to a perfect prediction. Points concentrated around 0 on both axes mean that the observed and/or predicted variation is globally small (mean effect close to 0 and high inter-event variability).

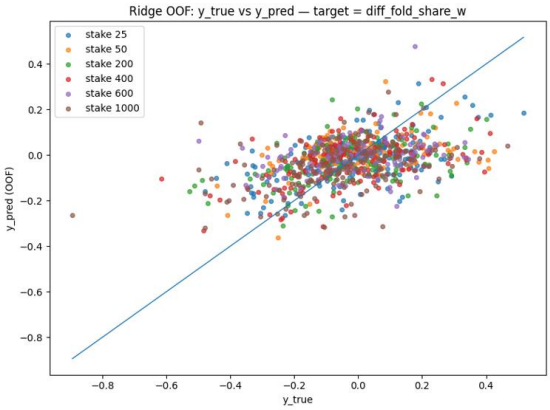


Figure 7 — Ridge OOF: y_true vs y_pred (target diff_fold_share_w, points colored by stake).

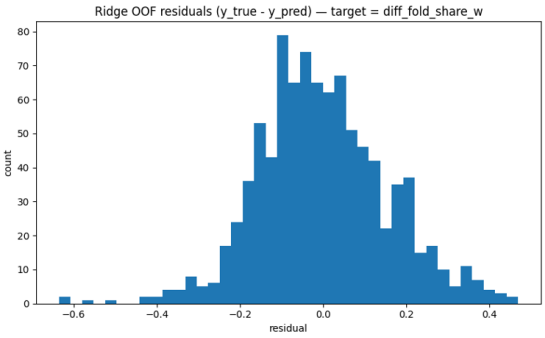


Figure 8 — Ridge OOF: residuals (y_true - y_pred) (target diff_fold_share_w).

Figure 9 — Ridge OOF: y_true vs y_pred (target diff_median_size_bb, points colored by stake).

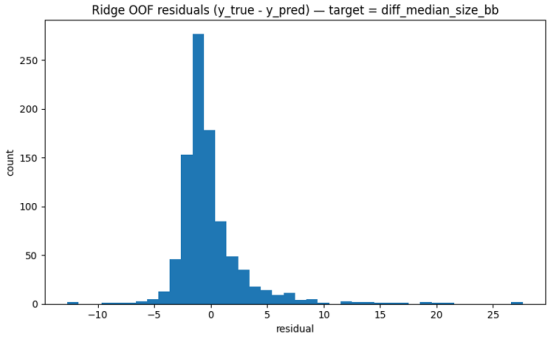


Figure 10 — Ridge OOF: residuals (y_true - y_pred) (target diff_median_size_bb).

For diff_fold_share_w (Figures 7–8), the cloud is tightly concentrated around 0 and the range of y_pred is narrower than that of y_true: the model tends to predict moderate changes and underestimates extreme variations, which is typical of a penalized regression when the

signal is weak or noisy. This is explained by two mechanisms: (i) the Ridge Regression (Ridge regression) penalty “shrinks” coefficients toward 0 (shrinkage), which mechanically reduces the amplitude of predictions; (ii) the available features mainly describe the “average” pre-event profile and capture rare, contextual events poorly (table dynamics, specific runouts), so extreme variations are not very predictable and are pulled toward a prediction close to 0.

For diff_median_size_bb (Figures 9–10), alignment with the diagonal is more visible (consistent with higher R^2), but we observe compression of predicted values and a few marked outliers in y_true (e.g., large sizing decreases), which generate asymmetry and a longer tail in the residual distribution. This configuration is expected when the target variable has heavy tails: a few extreme observations dominate the error but remain difficult to explain with aggregated variables (pre-event means/medians). Ridge then favors a “stable” solution that optimizes overall error at the expense of predicting rare cases; consequently, extremes are under-predicted and result in larger residuals and a more asymmetric distribution.

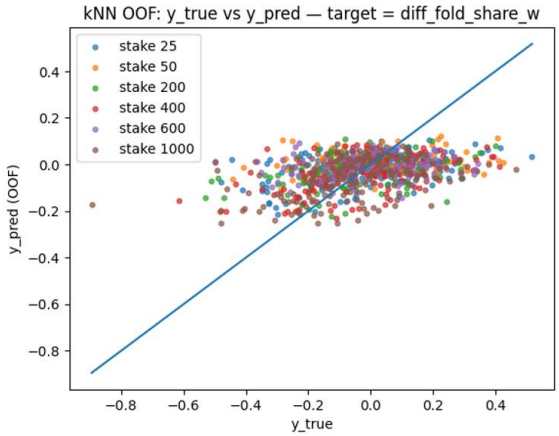


Figure 11 — k-Nearest Neighbors (kNN) OOF: y_true vs y_pred (target diff_fold_share_w).

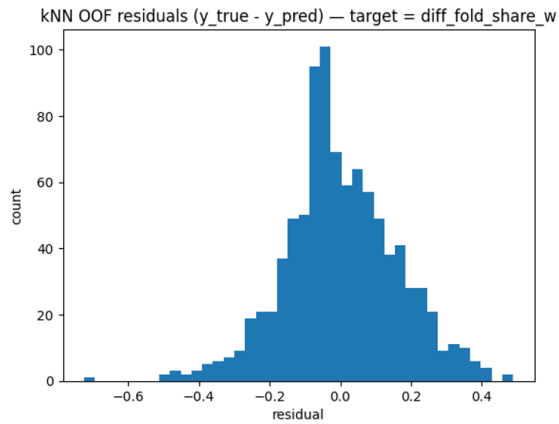


Figure 12 — k-Nearest Neighbors (kNN) OOF: residuals (target diff_fold_share_w).

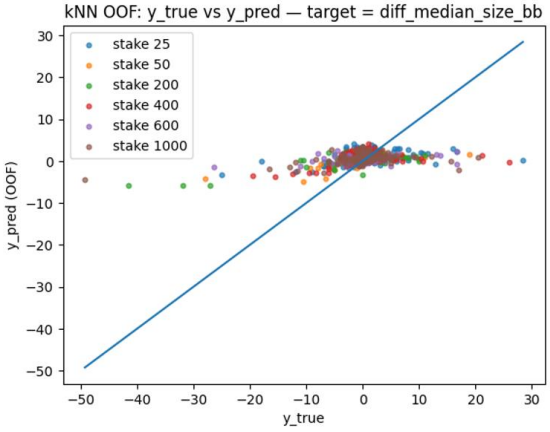


Figure 13 — k-Nearest Neighbors (kNN) OOF: y_true vs y_pred (target diff_median_size_bb).

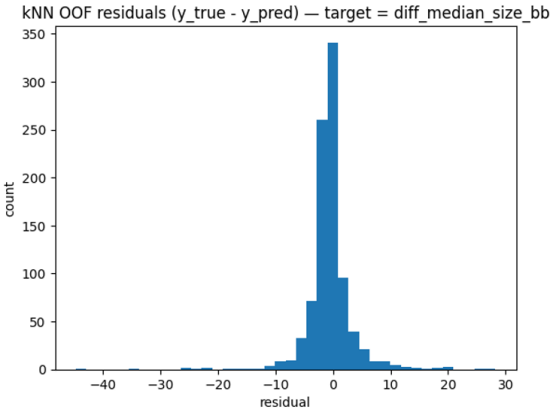


Figure 14 — k-Nearest Neighbors (kNN) OOF: residuals (target diff_median_size_bb).

K-Nearest Neighbors (kNN) produces very “averaged” predictions: y_pred often stays close to 0, with reduced amplitude compared with y_true. This reflects under-learning of extreme variations and explains generally lower performance on this task.

How can we explain this behavior? First, kNN is a local model: a prediction is essentially the mean or the median of the targets observed among the *k* nearest neighbors in the space of features. However, in our case, the “change after a big loss” is a weak and highly variable signal: for many similar pre-event profiles, post-event reactions can be opposite (some increase, others decrease). The neighbors’ average therefore naturally tends toward 0, which compresses the amplitude of the predictions.

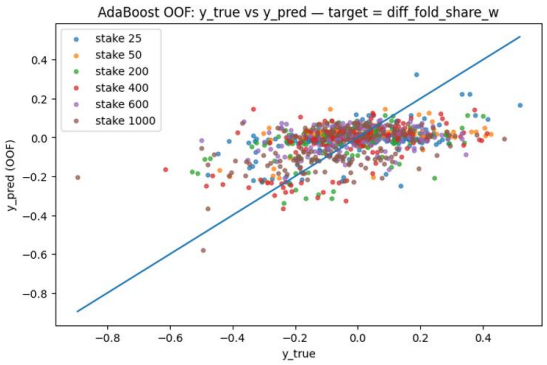


Figure 15 — AdaBoost (AdaBoost) — Boosting (boosting) OOF: y_true vs y_pred (target diff_fold_share_w).

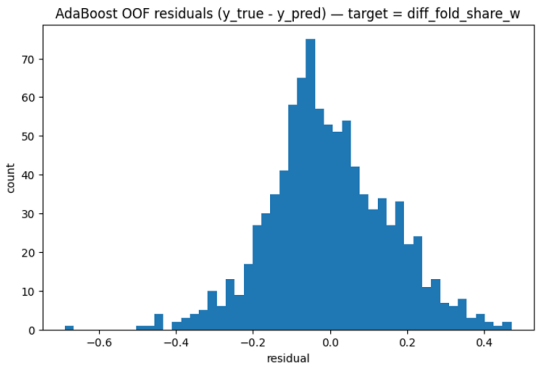


Figure 16 — AdaBoost (AdaBoost) — Boosting (boosting) OOF: residuals (target diff_fold_share_w).

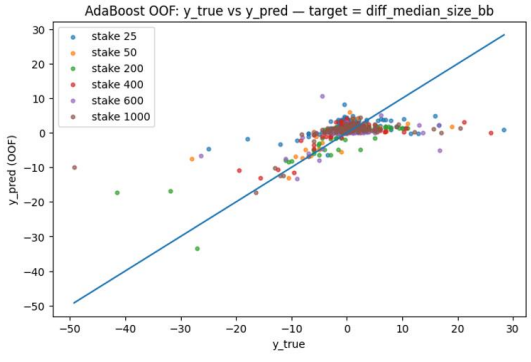


Figure 17 — AdaBoost (AdaBoost) — Boosting (boosting) OOF: y_true vs y_pred (target diff_median_size_bb).

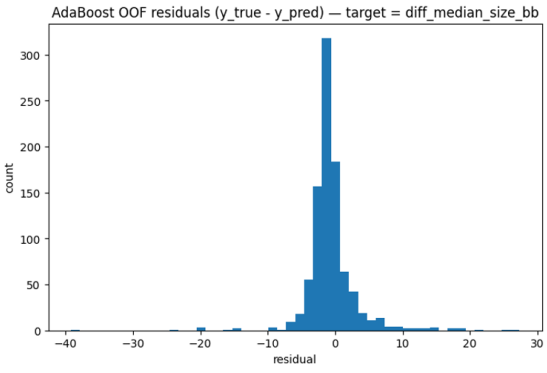


Figure 18 — AdaBoost (AdaBoost) — Boosting (boosting) OOF: residuals (target diff_median_size_bb).

AdaBoost can capture non-linearities and sometimes improve the score on certain stakes, but predictions remain globally compressed, especially on diff_fold_share_w.

How can we explain this behavior? On the one hand, AdaBoost relies on a sum of weak learners (often very shallow trees) that progressively correct residual errors. This structure makes it possible to model non-linear effects and simple interactions (for example, thresholds on pre-event sizing or on the number of hands), which can help when the relationship between features and target is not well described by a linear model. On the other hand, when the signal is weak and the target very noisy (typical case of diff_fold_share_w), AdaBoost tends to learn cautious rules that mainly explain “common” variations, and it struggles to predict extreme cases: these are rare, heterogeneous, and often linked to unobserved factors (table dynamics, runouts, momentary mental state), which are difficult to generalize.

VII. Conclusion

This project aimed to quantify and characterize potential tilt following a “big loss” in PokerStars cash games, by analyzing changes across several behavioral dimensions (tendency to fold, bet sizing, aggressiveness) and by testing the predictability of these changes from the pre-event player profile. The analysis was conducted stake-by-stake, to avoid aggregating potentially very different player populations and table dynamics.

From a descriptive and inferential standpoint, the main conclusion is that most average changes are close to zero: for many stakes and metrics, the 95% confidence intervals (95% CIs) overlap 0 and p-values are not significant, which is consistent with a weak average effect and/or strong inter-player heterogeneity. A few signals nevertheless stand out: for fold share, some stakes (notably 200 and 1000) show a statistically significant average decrease after the event, suggesting a more “sticky” behavior (players fold less) in these segments. For sizing, the median highlights at stake 25 a significant but borderline increase (+0.58 BB, p = 0.0476), with a 95% CI whose lower bound is very close to 0, supporting a small but potentially systematic effect. By contrast, aggressiveness (Bet/Raise Share) does not show any robust change: even when a trend appears (e.g., a slight increase at stake 1000), it remains inconclusive given the 95% CIs and p-values.

From a Machine Learning perspective, the ability to predict post-event changes remains overall modest, especially for diff_fold_share_w, where models produce predictions tightly concentrated around 0 and underestimate extreme variations. This is consistent with a weak and noisy signal and with the fact that key drivers of tilt (table context, runouts, opponent dynamics, rare decisions)

are not directly encoded in the pre-event aggregated features. Performance is generally better on diff_median_size_bb, suggesting that post-event sizing contains a slightly more structured signal than the propensity to fold. Ridge Regression (régression Ridge) provides a stable but compressed baseline (shrinkage), k-Nearest Neighbors (kNN) strongly “averages” cases and struggles to extrapolate extremes, while AdaBoost (AdaBoost) — Boosting (boosting) captures some non-linearities but still shows notable prediction compression, especially for the fold-related target.

In summary, the results suggest that tilt, as captured by these aggregated metrics, is often small on average, but may be present in a targeted way for certain stakes and certain aspects of play (notably folding and sizing). The predictive models confirm that the share of the phenomenon that can be explained from the pre-event profile is limited, reinforcing the idea that tilt is a highly contextual and heterogeneous phenomenon, more visible in specific subpopulations or individual cases than in an overall mean effect.

References and appendix

Kim, J. (2024). A Dataset of Poker Hand Histories [Data set]. Zenodo. https://doi.org/10.5281/zenodo.13997158

Kahneman, D. and Tversky, A. (1979). *Prospect Theory: an Analysis of Decision under Risk*. [online] Available at: https://web.mit.edu/curhan/www/docs/Articles/15341_Readings/Behavioral_Decision_Theory/Kahneman_Tversky_1979_Prospect_theory.pdf.

Zhang, K. and Clark, L. (2020). Loss-chasing in gambling behaviour: neurocognitive and behavioural economic perspectives. *Current Opinion in Behavioral Sciences*, [online] 31, pp.1–7. doi:https://doi.org/10.1016/j.cobeha.2019.10.006.

Moreau, A., Delieuvin, J., Chabrol, H. and Chauchard, E. (2017). Online Poker Tilt Scale (OPTS): creation and validation of a tilt assessment in a French population. *International Gambling Studies*, 17(2), pp.205–218. doi:https://doi.org/10.1080/14459795.2017.1321680.

Thaler, R.H. and Johnson, E.J. (1990). Gambling with the House Money and Trying to Break Even: The Effects of Prior Outcomes on Risky Choice. *Management Science*, [online] 36(6), pp.643–660. Available at: https://www.jstor.org/stable/2631898.

ChatGPT (OpenAI) — assistance with report writing/editing, structuring the Results section, and refining explanations and interpretations based on outputs produced in the notebooks.

Python (Google Colab) — data processing, exploratory analysis, statistical testing, and model training/evaluation.