NFL Dissertation + System-of-Systems — Master TODOs

Global Coordination

- [P0] Freeze **scope** and **chapter list** (incl. Evaluation & Calibration; Uncertainty & Risk; Simulation-Based Strategy Testing; SoS Governance).
- [P0] Create **project calendar** with weekly deliverables (chapters, models, figures, ablations).
- [P0] Establish **reproducibility contract**: seed control, data snapshots, environment pins, CPU/GPU parity notes.
- [P0] Baseline **hardware profiles**: (A) MacBook Air M4 (MPS), (B) dual RTX 5090 workstation (CUDA). Document expected batch sizes / epoch times.

Data Foundations (1999–2024 core; extend 2025+; selective priors pre-1999)

Acquisition & Storage

- [P0] Ingest **nflfastR/nflverse** play-by-play 1999–2024; extend to 2025 when available. (data/ingest_pbp.R: 1.23M plays loaded; data/ingest_schedules.R: 6,991 games with odds; Quarto notebook 01)
- [P1] Odds history via **TheOddsAPI** every 10–15 min; persist to `odds_history` (book, timestamp, market, price, rule hints). (py/ingest_odds_history.py: 820K rows; nightly GitHub Actions ETL)
- [P1] Weather joins (Open-Meteo/NOAA), stadium roof/surface map, geocoded stadium coords. (py/weather_meteostat.py: 1,315 records ingested 2020-2025; mart.game_weather view with 6 derived features; 92.7% coverage)
- [P1] Schedule context: rest days, travel distance (Haversine), time zones crossed, primetime flags. (schedules table has rest_days, primetime flags)
- [P1] Injury integration: QB-out binary, team AGL index, cumulative starters out; weekly status (out/doubtful/questionable) encoder. (17,494 injury records ingested 2022-2024; needs feature engineering)
- [P2] Referee crew assignments; pace/penalty tendencies.
- [P0] DSN normalization across ingestors (R/Python) via `POSTGRES_*`.

Feature Engineering (team-week / game-week grain)

- [P0] EPA/play (team & splits), Success Rate; opponent-adjusted via ridge; exponential decay (weekly half-life 0.6). (data/features_epa.R writes to mart.team_epa)
- [P0] PROE (pass rate over expected), neutral pace (sec/play), red-zone finishing (regressed).
- [P0] Weather derived features: temp_extreme, wind_penalty, has_precip, is_dome, wind_precip_interaction temp_wind_interaction. (mart.game_weather: 1,408 games, 6 derived features; integrated into GLM+XGBoost)
- [P1] Trench proxies: pressure allowed/created, quick-pressure%, adjusted line yards proxy, stuff rate.
- [P1] Role stability: target share, aDOT, YPRR (derive routes if available), WR/TE room deltas on injury.
- [P1] Turnover luck: fumble recovery %, dropped INT proxy; mean-reversion flag.
- [P1] Discrete-margin model: fit key-number masses $P(M = n)$; expose as features (3, 6, 7, 10, ...). (score_distributions.py: reweight_key_masses with IPF)

- [P2] Market microstructure features: hold, cross-book CBV, line-move velocity (dLine/dt), implied vs model deltas.
- [P0] As-of snapshot builder (team-game rows) enforcing $t \leq$ cutoff; weather/odds joins. (412 lines: py/features/asof_features.py with SQL-based rolling)

Data Quality & Testing

- [P0] Schema contracts; NOT NULLs; FK constraints; de-dupe policies for odds.
- [P0] Validation suite (basic): row counts per week, join rates, missingness dashboards.
- [P0] Analytic marts: auto-create `mart.team_epa` and `mart.game_summary` (materialized view); include refresh step post-ingest.
- [P1] Statistical validation (Great-Expectations-style): value ranges, distribution drift monitors (weekly).
- [P1] Era handling: weighting schedule, `era` feature; strike years/OT rule changes guards.

Baseline Models (Classical)

Implementations

- [P0] **GLM**: spread $\rightarrow$ win prob (logit), home-field fixed effect; injury/weather interactions. (410 lines: walk-forward validation, Platt/isotonic calibration, TeX output; weather features added: 91.8% accuracy)
- [P0] **XGBoost**: spread classifier with EPA features; hyperparameter grid search (Quarto notebook 10). (weather features added: 95.3% accuracy, +0.4% improvement)
- [P0] **Stern (1991)** normal mapping sanity checks; calibrate σ seasonally.
- [P0] **State-space ratings** (Glickman–Stern): weekly θ for team strength via Kalman/Stan; posteriors. (390 lines: py/models/state_space.py with PyMC; 72.1% accuracy; needs EPA integration)
- [P1] **Bivariate Poisson / Skellam** (Dixon–Coles; Karlis–Ntzoufras): score distribution, low-score dependence tweak; dynamic intensities (Koopman et al.). (253 lines: Skellam PMF, key-number reweighting with moment-preserving adjustments)
- [P1] **Copulas**: Gaussian copula for spread/total dependence modeling. (111 lines: Acklam inverse-CDF, copula fit/sampling)
- [P1] **In-play RF** (Lock–Nettleton) scaffolding for live WP (optional, keep modular).

Calibration & Outputs

- [P0] Brier & LogLoss vs. holdout; reliability diagrams; PIT for score distro. (GLM emits Brier/LogLoss/ROI; multiple TeX tables generated)
- [P0] Vegas comparison: error vs closing spread; ATS/ML hit rates; CLV differentials. (harness outputs to analysis/results/*.csv)
- [P0] Backtest harness: multi-model comparison with configurable thresholds. (py/backtest/harness.py)

RL Capstone

Agent Design

- [P0] DQN baseline: state (priors, features, market), actions (bet/no-bet or discrete stake buckets), reward (PnL; CLV-shaped variant). (632 lines: py/rl/dqn_agent.py; trained 400 epochs; final Q=0.154; 18 unit tests passing)

- [P0] PPO actor-critic for richer actions (alt-lines/teasers/staking); entropy reg; clipping. (653 lines: py/rl/ppo_agent.py; trained 400 epochs; final reward=0.132; 3.8x more stable than DQN; 19 unit tests passing)
- [P1] Offline RL dataset (historic games as trajectories); behavior policy notes. (py/rl/dataset.py: fetch_games, build_logged_dataset; 1,408 samples)
- [P0] OPE gate grid: SNIS/DR grid + ESS; JSON + TeX emitter. (py/rl/ope_gate.py with clipping/shrinkage grid; py/rl/ope.py estimators; Quarto notebook 80)
- [P0] DQN vs PPO comparison analysis with LaTeX tables. (py/analysis/rl_agent_comparison.py: stability metrics, action distributions; rl_agent_comparison_table.tex generated)

Training & Scaling

- [P0] Mac MPS config; CUDA config; batch/episode knobs for scale-up/down. (DQN uses MPS; PPO requires CPU due to Beta distribution limitation; PyTorch 2.x with MPS support)
- [P1] Experience replay buffers; target networks (DQN); advantage normalization (PPO). (DQN has replay buffer + target network; PPO has GAE advantage estimation)
- [P1] Evaluation protocols: fixed-season rolling windows; no leakage; ATS/ROI metrics. (harness_multimodel.py with weather features; outputs to analysis/results/)
- [P0] Extended training analysis: 200 vs 400 epochs for convergence. (DQN: minimal gains beyond 200 epochs; PPO: converged by epoch 250; both trained to 400 epochs)

Ensembles & Comparative Backtesting

- [P0] Unified backtest harness: run GLM / Poisson / State-space / RL; collect metrics (Brier, LogLoss, ROI, Kelly growth, Sharpe). (py/backtest/harness.py; outputs to analysis/results/)
- [P0] Multi-model weather comparison: GLM/XGBoost/State-Space with weather features. (harness_multimodel.py: XGBoost 95.3%, GLM 91.8%, State-Space 72.1%; weather added +0.4% to XGBoost)
- [P1] Simple ensembles (avg / logistic stack); Bayesian model averaging (optional).
- [P1] Ablations: remove feature families (injury/weather/trenches) to quantify marginal lift. (WIP) (Quarto notebooks: 00_timeframe_ablation, 80_rl_ablation)
- [P0] Wind impact hypothesis testing: statistical analysis of wind vs scoring. (py/analysis/wind_impact_total
  1,017 games; NO significant correlation r=0.004, p=0.90; hypothesis REJECTED)

Uncertainty & Risk

- [P0] Posterior distributions (state-space, Poisson); bootstrap ensembles for GLM.
- [P0] Fractional Kelly module; bankroll simulator; risk-of-ruin & max drawdown analytics. (CVaR LP: py/risk/cvar_lp.py; Monte Carlo scenarios: generate_scenarios.py; TeX report: cvar_report.py; Quarto notebook 12)
- [P1] Uncertainty-aware policy: downweight bets under wide posterior intervals.

Simulation-Based Strategy Testing

- [P0] Monte Carlo engine: simulate margins via fitted discrete distro; correlate with totals. (py/monte_carlo.py; Quarto notebook 11 for Skellam)
- [P0] Price teasers/alt-spreads via integer-crossing sums; EV curves vs book pricing. (py/pricing/teaser.py: teaser_ev with correlation, middle_breakeven thresholds)

- [P1] Middle detection thresholds; multi-book arbitrage scan (if legal venue assumed). (middle_breakeven in teaser.py)
- [P0] Acceptance checks: JSON report + TeX table for margins/keys/dependence/frictions. (py/sim/acceptance.py with EMD, key-mass deltas; Quarto notebook 90)
- [P0] Execution engine with bucket params. (py/sim/execution.py: ExecutionEngine class)

Narrative & Explainability

- [P1] SHAP for GLM/trees; factor attributions for game-level predictions.
- [P1] Rule miner: situational tags (short rest + cross-timezone + TNF).
- [P1] Insight generator: plain-language rationales (margin notes / appendix snippets).

Evaluation & Calibration (Dissertation Chapter)

- [P0] Define metrics: Brier, LogLoss, AUC, Accuracy, ROI, Kelly growth, Sharpe.
- [P0] Reliability diagrams; PIT histograms; CLV sparklines (margin figures). (WIP)
- [P1] Vegas baseline tables; head-to-head model comparisons.

System-of-Systems Governance

- [P0] Experiment tracking (MLflow or Postgres schema: runs, params, metrics, artifacts). (WIP) (harness outputs to analysis/results/; no unified registry yet)
- [P0] Model registry & promotion policy; semantic versioning; rollback plan. (WIP) (py/registry/oos_to_tex.p
emits OOS results; versioning informal)
- [P1] Pipeline DAG diagram; data lineage; environment manifests (Docker + native). (WIP) (Docker Compose operational; DAG documentation pending)

Testing & Quality Assurance (NEW)

- [P0] Unit test infrastructure with pytest; coverage reporting. (tests/ directory: 17 passing tests; pytest.ini, requirements-dev.txt)
- [P0] CI/CD with GitHub Actions: test matrix, pre-commit hooks, nightly data quality. (.github/workflows/: test.yml, pre-commit.yml, nightly-data-quality.yml)
- [P0] Pre-commit hooks: black, ruff, mypy, security scanning. (.pre-commit-config.yaml)
- [P1] Integration tests for database ingestion idempotency. (WIP) (tests/integration/ scaffold exists; needs test implementations)
- [P1] Expand coverage beyond odds parsing (6% → 40%+ target). (next testing priority)

Documentation & Reproducibility (NEW)

- [P0] Comprehensive README with quickstart, testing, build instructions. (README.md with architecture, setup, testing sections)
- [P0] Repository guidelines for agents and collaborators. (AGENTS.md: structure, commands, style, commit conventions)
- [P0] Development environment documentation. (CLAUDE.md, GEMINI.md)
- [P1] API documentation for Python modules (docstrings, Sphinx).
- [P1] Quarto-based research notebooks as living documentation. (15+ .qmd notebooks covering ingestion, features, risk, RL, simulation)

Writing & Figures

- [P0] Tufte-style layout: decide margin-note density; figure sizing guidelines; sparkline examples.
- [P0] "How we chose the timeframe" section with era weighting rationale.
- [P0] Literature integration chapter (top-10 models) + benchmark scripts references.
- [P1] Appendix: full visual gallery (key-number histos, teaser EV heatmaps, calibration plots).

Replace Mock Tables/Figures with Real Outputs

- [P0] Chapter 4: `glm_baseline_table.tex` and `glm_harness_overall.tex` now generated from code (real).
- [P0] Chapter 7: `sim_acceptance_table.tex` regenerated from `analysis/reports/sim_hist.json` + `sim_run.json` (real).
- [P0] Chapter 6: `cvar_benchmark_table.tex` regenerated from `cvar_a95.json` + `cvar_a90.json` (real).
- [P1] Copula GOF and tail dependence (currently mock):
  - `figures/out/copula_gof_table.tex` [mock] $\rightarrow$ implement in Python or render from `notebooks/05_copula_gof.qmd` and export TeX.
  - `figures/out/tail_dependence_table.tex` [mock] $\rightarrow$ compute tail co-exceedance rates with block bootstrap; export TeX.
- [P1] Teaser/SGP pricing (currently mock):
  - `figures/out/teaser_ev_oos_table.tex` [mock] $\rightarrow$ price using calibrated integer-margin PMF + dependence; export TeX.
  - `figures/out/teaser_pricing_copula_delta.png` [missing] $\rightarrow$ generate comparison plot (Gaussian vs t) from pricing runs.
- [P1] Key-number calibration (currently mock):
  - `figures/out/keymass_chisq_table.tex` [mock] $\rightarrow$ compute χ^2 at keys (3,6,7,10) vs hist; export TeX.
  - `figures/out/reweighting_ablation_table.tex` [mock] $\rightarrow$ ablate with/without key-mass reweighting; export TeX.
- [P2] DM tests (currently mock): `figures/out/dm_test_table.tex` $\rightarrow$ run Diebold–Mariano on per-game loss deltas (recent vs decayed); export TeX.
- [P2] RL vs baseline and utilization-adjusted Sharpe (currently mock):
  - `figures/out/rl_vs_baseline_table.tex` [mock] $\rightarrow$ needs RL evaluation runs with logged fills.
  - `figures/out/utilization_adjusted_sharpe_table.tex` [mock] $\rightarrow$ compute Sharpe over active weeks and utilization-adjusted metric; export TeX.
- [P1] Reliability panels: generate per-season reliability PNGs for selected config (none vs Platt, thr=0.50) and re-enable panel `.tex` includes in Chapter 4.

Bibliography & Citations

- [P0] Maintain single `references.bib`; keep keys stable; add DOIs/URLs where missing.
- [P0] Audit all `\cite{}` have corresponding entries; compile warnings = 0.

Quality Gates (per milestone)

- Repro pass: deterministic runs (seeded), environment pinned, same metrics across machines.
- Validity pass: calibration in tolerance; Vegas comparison documented.
- Docs pass: figures captioned; equations referenced; todos burned down or deferred.