valuation/backtesting and a simple app UX so you can show off results in interviews/portfolio.

**TL;DR**

Build a match-level probability model (e.g., Poisson / bivariate Poisson or a gradient-boosted classifier calibrated to probabilities), then run Monte-Carlo season simulations from those match probabilities to estimate *champion probabilities*. Use free historical CSVs + APIs (bookmakers odds, xG, injuries, transfers) as features. Host the Streamlit UI on Hugging Face Spaces or Streamlit Community Cloud. (Sources for key data & deployment options below.) [Football Data](https://www.football-data.co.uk/?utm_source=chatgpt.com)[Football Data](https://www.football-data.org/documentation/api?utm_source=chatgpt.com)[understat.readthedocs.io](https://understat.readthedocs.io/en/latest/index.html?utm_source=chatgpt.com)[The Odds API](https://the-odds-api.com/?utm_source=chatgpt.com)[Hugging Face](https://huggingface.co/docs/hub/en/spaces?utm_source=chatgpt.com)

**Where to get data (good, free starting points)**

* Historical match CSVs (results + basic stats / odds): **football-data.co.uk** (downloadable CSVs going many seasons — great for backtesting). [Football Data](https://www.football-data.co.uk/?utm_source=chatgpt.com)
* League / fixtures / standings APIs: **football-data.org** (REST API with fixtures/standings; requires API key). [Football Data](https://www.football-data.org/documentation/api?utm_source=chatgpt.com)
* Rich stats + commercial API (free tier available): **API-Football / API-Sports** — fixtures, lineups, events, injuries, odds (free tier limits apply). [api-football+1](https://www.api-football.com/documentation-v3?utm_source=chatgpt.com)
* Expected Goals (xG) & advanced metrics: **Understat** (commonly used; Python wrappers exist to pull xG per match/team/player). [understat.readthedocs.io](https://understat.readthedocs.io/en/latest/index.html?utm_source=chatgpt.com)[Understat](https://understat.com/?utm_source=chatgpt.com)
* Open event datasets (research / deep features): **StatsBomb open-data** (useful for past seasons / experiments). [GitHub](https://github.com/statsbomb/open-data?utm_source=chatgpt.com)
* Bookmaker odds (powerful prior features): **The Odds API / the-odds-api** or similar odds providers (free tiers exist). Odds → implied probabilities are great baseline predictors. [The Odds API](https://the-odds-api.com/?utm_source=chatgpt.com)
* Fantasy / player-level endpoints: **Fantasy Premier League (FPL) public endpoints** (fixtures, player form) — useful for player availability / form features. [GitHub](https://github.com/jeppe-smith/fpl-api?utm_source=chatgpt.com)
* Transfer market / market value (optional): Transfermarkt (scraping) or Kaggle snapshots for squad values (useful proxy for “team strength”, but verify TOS). [Medium](https://footballdotpy.medium.com/scraping-premier-league-squad-valuations-from-transfermarkt-db8306856d6b?utm_source=chatgpt.com)[Kaggle](https://www.kaggle.com/datasets/evangower/premier-league-matches-19922022?utm_source=chatgpt.com)

Quick notes: many free APIs have rate limits / licensing. Check each provider’s TOS before heavy use or commercialising. [api-football](https://www.api-football.com/pricing?utm_source=chatgpt.com)[Postman](https://www.postman.com/api-noob/football-data-org-apis/documentation/yjgfm4j/football-data-org-v4?utm_source=chatgpt.com)

**What to predict (suggestions)**

Primary (showcase project):

* **Probability that each team will win the coming Premier League season** — expressed as percentage from Monte-Carlo season simulation (best for recruiting demos).

Secondary / extras:

* Match outcome probabilities (H/D/A) and predicted scorelines.
* Final points distribution / probability of top-4, relegation zone.
* Top scorer and expected points for teams.
* Live updating model that ingests new fixtures/odds and updates champion probabilities.

Why simulation? Forecasting *match-by-match* probabilities then simulating the whole season (many times) yields calibrated champion probabilities and lets you show uncertainty (the same approach FiveThirtyEight uses). [ABC News](https://fivethirtyeight.com/features/how-our-club-soccer-projections-work/?utm_source=chatgpt.com)

**Modelling approach (recommended stack & techniques)**

1. **Baseline (fast MVP):** Use a bookmaker-odds baseline (implied probs) + simple Elo rating or Poisson model. Odds are a very strong prior; combine them with historical model outputs.
2. **Statistical goal model:** Poisson or **bivariate Poisson** to predict goals (Maher / Dixon/Coles tradition). From predicted goal rates you compute match outcome probabilities. (Lots of literature and good baseline performance.) [Open Access LMU](https://epub.ub.uni-muenchen.de/29028/1/TR_EM2016.pdf?utm_source=chatgpt.com)[ResearchGate](https://www.researchgate.net/profile/Phil-Scarf/publication/288065035_Modelling_the_outcomes_of_association_football_matches/links/5e301f3192851c7f7f0579df/Modelling-the-outcomes-of-association-football-matches.pdf?utm_source=chatgpt.com)
3. **Machine-learning upgrade:** Train XGBoost / LightGBM / CatBoost to predict match outcome probabilities (features below). Calibrate probabilistic outputs (Platt/isotonic) and/or stack with Poisson/Elo/odds.
4. **Season simulation:** For each remaining fixture, sample match results from predicted probability distributions; simulate the whole season N times (e.g., 10k) and compute how often each team finishes 1st. FiveThirtyEight style. [ABC News](https://fivethirtyeight.com/features/how-our-club-soccer-projections-work/?utm_source=chatgpt.com)
5. **Optional Bayesian version:** Build a hierarchical Poisson or dynamic model (PyMC/Pyro) for uncertainty quantification.

**Useful features to engineer**

Match-level:

* Home/Away indicator; days-rest; matchday (round number).
* Bookmaker implied win/draw/lose probabilities (powerful). [The Odds API](https://the-odds-api.com/?utm_source=chatgpt.com)
* Team attack/defense strengths (rolling averages, last-N form).
* xG for/against (Understat/FBref). [understat.readthedocs.io](https://understat.readthedocs.io/en/latest/index.html?utm_source=chatgpt.com)[FBref.com](https://fbref.com/en/comps/9/Premier-League-Stats?utm_source=chatgpt.com)
* Elo or SPI rating (online-updating rating). [ABC News](https://fivethirtyeight.com/methodology/how-our-club-soccer-predictions-work/?utm_source=chatgpt.com)
* Head-to-head recent results, travel distance, referee (if you want).  
  Team-level (seasonal / static):
* Squad market value or net spend (Transfermarkt / scraped/Kaggle snapshots). [Medium](https://footballdotpy.medium.com/scraping-premier-league-squad-valuations-from-transfermarkt-db8306856d6b?utm_source=chatgpt.com)
* Manager tenure / changes, injuries & suspensions (from APIs). [api-football](https://www.api-football.com/documentation-v3?utm_source=chatgpt.com)

**Step-by-step implementation plan (concrete)**

I’ll assume you’ll use Python (pandas, scikit-learn, xgboost/lightgbm, statsmodels/pyMC optional). Here’s a reproducible pipeline.

1. **Project scaffold & repo**
   * Create GitHub repo with README, requirements.txt, a LICENSE, and folders: data/, notebooks/, src/, app/, models/.
   * Optional: Dockerfile for reproducibility.
2. **Data collection (ETL)**
   * Pull historical CSVs from football-data.co.uk for backtesting. [Football Data](https://www.football-data.co.uk/?utm_source=chatgpt.com)
   * Register for API keys: football-data.org and / or API-Football; set up small wrapper functions to fetch fixtures, standings, odds, injuries. [Football Data](https://www.football-data.org/documentation/api?utm_source=chatgpt.com)[api-football](https://www.api-football.com/documentation-v3?utm_source=chatgpt.com)
   * Scrape Understat for xG (or use Python wrapper) and FBref for extra stats. Cache snapshots so backtests are reproducible. [understat.readthedocs.io](https://understat.readthedocs.io/en/latest/index.html?utm_source=chatgpt.com)[FBref.com](https://fbref.com/en/comps/9/Premier-League-Stats?utm_source=chatgpt.com)
3. **Exploratory Data Analysis**
   * Basic league stats, seasonality, home advantage, xG vs goals sanity checks. Visualize distributions, correlations, missing values.
4. **Baseline models**
   * Build an *odds-only* baseline (convert odds → implied probs, normalize, measure Brier/log loss).
   * Implement Elo (or use existing code) and evaluate match prediction performance. [ABC News](https://fivethirtyeight.com/methodology/how-our-club-soccer-predictions-work/?utm_source=chatgpt.com)
5. **Statistical goal model (MVP predictive engine)**
   * Fit independent Poisson or **bivariate Poisson** models for home/away goals (Maher/Dixon approaches). Evaluate by log-likelihood & accuracy of predicted score distributions. [Open Access LMU](https://epub.ub.uni-muenchen.de/29028/1/TR_EM2016.pdf?utm_source=chatgpt.com)
6. **ML model(s)**
   * Train XGBoost classifier/regressor on match features to predict outcome probs and/or goals. Use cross-validation (time-series CV / rolling window) not random split. Calibrate probabilities (isotonic/Platt).
   * Feature importance & SHAP to explain predictions (great for interviews).
7. **Season simulation**
   * For the *coming season* (or remaining fixtures), for each fixture compute match probs (from ensemble of models + odds). Run N simulations (10k) sampling match outcomes (sample scores from Poisson or H/D/A from probabilities), compute final tables, and estimate champion probability distribution. (This is the main deliverable.) [ABC News](https://fivethirtyeight.com/features/how-our-club-soccer-projections-work/?utm_source=chatgpt.com)
8. **Backtesting / validation**
   * Backtest on past seasons: train model using only data available before the season and measure how well simulated champion probabilities match actual champions (calibration, Brier score).
   * Evaluate match-level metrics (log loss, Brier) and season-level calibration.
9. **Build Streamlit app (MVP)**
   * UI features: overall champion probabilities (bar + table), team pages (Elo time series, xG trends), interactive simulation sliders (e.g., “give Team X +3 market value”), match-by-match probability viewer, explainability panel (SHAP).
   * Hook the model code to a lightweight API or use the model serialized (joblib/pickle) and load in Streamlit.
10. **Deploy**

* Quick & free for public demos: **Hugging Face Spaces** (Streamlit supported) or **Streamlit Community Cloud** (easy GitHub integration). Hugging Face Spaces is especially friendly for ML demos and portfolio hosting. [Hugging Face](https://huggingface.co/docs/hub/en/spaces?utm_source=chatgpt.com)[Streamlit Docs](https://docs.streamlit.io/deploy/streamlit-community-cloud?utm_source=chatgpt.com)
* For more control: Render / Koyeb / Railway / Docker on cloud provider. (If you need GPU for heavy model inference, Hugging Face offers paid options.)

1. **Polish for portfolio**

* Clean README with methodology, data sources (cite & link), reproducible notebooks, sample results, and screenshots. Provide a short “how it works” explainer and a link to the live app.

**Folder / file sketch (good for interviews)**

bash

CopyEdit

/epl-predictor

/data # raw + processed

/notebooks # EDA + backtest

/src

/data # ETL scripts

/models # training, evaluation

/simulator # Monte Carlo season sim

/utils

/app # streamlit app files

requirements.txt

Dockerfile

README.md

**Evaluation metrics (what to show)**

* **Match level:** Log loss (cross-entropy), Brier score, calibration plots (reliability).
* **Season level:** Calibration of champion probabilities (e.g., if you predict 40% and simulate many seasons, do they win ~40%?). Backtest over several past seasons and show confusion (predicted top-1 vs actual).
* Visuals: probability bands, uncertainty ribbons, Elo time series, SHAP explanations.

**Quick implementation tips & libraries**

* Data & ETL: pandas, requests, soccerdata / worldfootballr (wrappers) — these help load FBref/football-data sources. [soccerdata.readthedocs.io](https://soccerdata.readthedocs.io/en/latest/datasources/MatchHistory.html?utm_source=chatgpt.com)[jaseziv.github.io](https://jaseziv.github.io/worldfootballR/articles/extract-fbref-data.html?utm_source=chatgpt.com)
* Models: scikit-learn, xgboost/lightgbm, statsmodels (Poisson), pymc/pyro for Bayesian.
* xG scraping: understat python package / wrapper. [GitHub](https://github.com/amosbastian/understat?utm_source=chatgpt.com)
* Streamlit app: streamlit + plotly/altair for plots; deploy to Hugging Face Spaces or Streamlit Community Cloud. [Hugging Face](https://huggingface.co/docs/hub/en/spaces?utm_source=chatgpt.com)[Streamlit Docs](https://docs.streamlit.io/deploy/streamlit-community-cloud?utm_source=chatgpt.com)

**Demo / MVP suggestions (so you can show impact quickly)**

* A single page Streamlit app with:
  + Top: **Champion probability dashboard** (bar chart, table).
  + Middle: **League simulation viewer** (run 10k sims, show distribution of finishing positions).
  + Bottom: **Team detail** (Elo + xG trend, form table) + explainability (SHAP).
* Add a “Compare to bookmakers” toggle (show implied bookie probability vs your model).

**Risks / caveats**

* Data licensing & scraping limits — always check providers’ terms (API quotas and commercial restrictions). [api-football](https://www.api-football.com/pricing?utm_source=chatgpt.com)[Postman](https://www.postman.com/api-noob/football-data-org-apis/documentation/yjgfm4j/football-data-org-v4?utm_source=chatgpt.com)
* Football is noisy; models should emphasize probabilistic outputs and uncertainty (rare to get a single-year prediction right). Use calibration & backtesting.

**Helpful links (quick)**

* football-data.co.uk (CSV historical results). [Football Data](https://www.football-data.co.uk/?utm_source=chatgpt.com)
* football-data.org (API docs). [Football Data](https://www.football-data.org/documentation/api?utm_source=chatgpt.com)
* API-Football docs (rich API). [api-football](https://www.api-football.com/documentation-v3?utm_source=chatgpt.com)
* Understat Python / docs (xG). [understat.readthedocs.io](https://understat.readthedocs.io/en/latest/index.html?utm_source=chatgpt.com)
* StatsBomb open data GitHub. [GitHub](https://github.com/statsbomb/open-data?utm_source=chatgpt.com)
* Odds API (bookmaker odds). [The Odds API](https://the-odds-api.com/?utm_source=chatgpt.com)
* Hugging Face Spaces (Streamlit hosting). [Hugging Face](https://huggingface.co/docs/hub/en/spaces?utm_source=chatgpt.com)
* Streamlit Community Cloud deploy docs. [Streamlit Docs](https://docs.streamlit.io/deploy/streamlit-community-cloud?utm_source=chatgpt.com)
* FiveThirtyEight method / Monte Carlo idea. [ABC News](https://fivethirtyeight.com/features/how-our-club-soccer-projections-work/?utm_source=chatgpt.com)
* Bivariate Poisson literature (Maher / modeling papers). [Open Access LMU](https://epub.ub.uni-muenchen.de/29028/1/TR_EM2016.pdf?utm_source=chatgpt.com)[ResearchGate](https://www.researchgate.net/profile/Phil-Scarf/publication/288065035_Modelling_the_outcomes_of_association_football_matches/links/5e301f3192851c7f7f0579df/Modelling-the-outcomes-of-association-football-matches.pdf?utm_source=chatgpt.com)