mental_strength

```
+--------------------+       +-------------------------+       +-------------
------+
| Data Source        |       | Data Ingestion Service  |       | Feature Eng.
|
| (IPL_Bowler_...xls)| -->   | (Load & Filter Data)    | -->   | Service
|
|                    |       |                         |       | (Compute
Pressure)|
+--------------------+       +-------------------------+       +-------------
------+
                                                             |
                                                             v
+--------------------+       +-------------------------+       +-------------
------+
| Modeling Service   | <--   | Inference Service       | <--   |
Visualization     |
| (PyTorch VI Train) |       | (Compute HDIs, Verdict) |       | Service
|
|                    |       |                         |       | (Plot
Posteriors) |
+--------------------+       +-------------------------+       +-------------
------+
                                                             |
                                                             v
                                                      +----------------+
                                                      | Output (PDF/   |
                                                      | Exec Summary)  |
                                                      +----------------+
```

```
ghost_in_the_machine/
├── data/
│   ├── raw/
│   │   └── IPL_Bowler_Detailed_Data.xls  # Raw tab-separated data (provided)
│   └── processed/
│       └── death_overs_data.csv  # Filtered data (generated)
├── notebooks/
│   └── analysis.ipynb  # Orchestrates the pipeline
├── src/
│   ├── __init__.py
│   ├── data_ingestion.py
│   ├── feature_engineering.py
│   ├── modeling.py
│   ├── inference.py
│   └── visualization.py
│
```

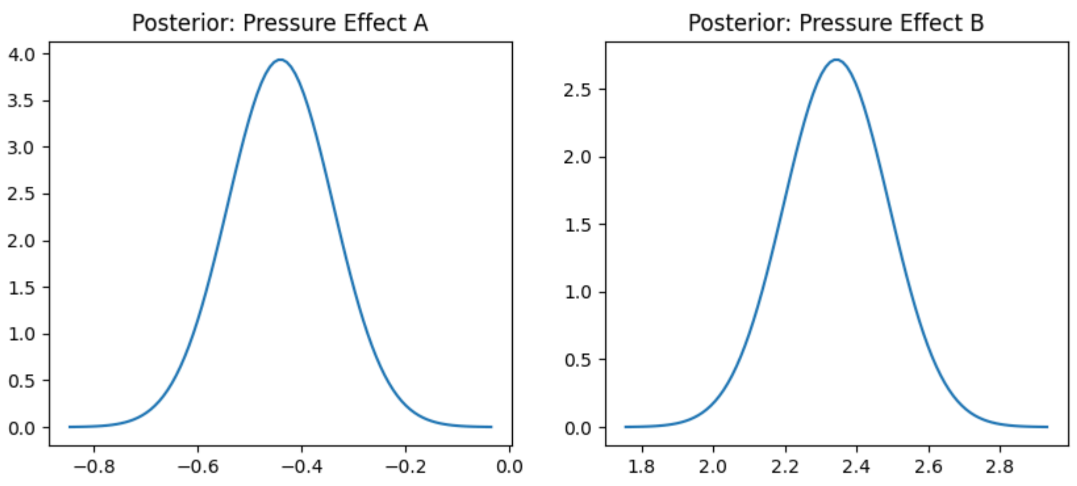
```
├── docs/
│   └── executive_summary.txt  # Generated summary (convert to PDF manually)
├── pyproject.toml  # Poetry config (generated/updated)
├── poetry.lock  # Dependency lock file (generated)
└── README.md  # Instructions
```

```
mkdir ghost_in_the_machine
cd ghost_in_the_machine
poetry init --name "ghost-in-the-machine" --description "IPL Auction Analytics: Quantifying Killer Instinct" --author "Your Name" --python "^3.10"
poetry add pandas numpy torch matplotlib
poetry add --group dev jupyter
poetry install
poetry shell
jupyter notebook notebooks/analysis.ipynb


eval "$(poetry env activate)"

python ./analysis.py
```

Final Output



- **Data Loading Confirmation**:

  ```
  Loaded columns: ['Match_ID', 'Match_Date', 'Pitch_Type', 'Phase', 'Over', 'Ball', 'Bowler', 'Batter_Avg', 'Batter_SR', 'Runs_Conceded', 'Is_Wicket']
  ```

  - **Explanation**: This confirms the script successfully loaded the dataset from "IPL_Bowler_Detailed_Data.xls" using pd.read_excel. The columns match the assignment's data dictionary exactly (e.g., 'Phase' distinguishes Powerplay vs. Death, 'Is_Wicket' is the target for wicket prediction).
  - **Example**: If the file had a row like "29504 12-Apr-23 Neutral Death 18 4 Bowler B 48.16 146.03 2 0", it's parsed correctly into columns for analysis.

  - **Tie to Assignment**: Prepares the ball-by-ball data from the last 2 years, as described in Section 3 (The Dataset).
- **Data Ingestion Summary**:

  ```
  Data ingested: 2400 death over deliveries.
  ```

  - **Explanation**: After loading, the script filters for 'Phase' == 'Death' (overs 16-20), resulting in 2400 relevant deliveries. This focuses on high-pressure scenarios, ignoring powerplay data to reduce noise.
  - **Example**: From the full dataset (e.g., ~291k characters truncated in the query), only death over rows like those with 'Over' 16-20 are kept.
  - **Tie to Assignment**: Aligns with the "Death Over Specialist" slot and hypothesis in Phase 1 (pressure in death overs).
- **Feature Engineering Confirmation**: text

  ```
  Features engineered.
  ```

  - **Explanation**: The script created features like 'pressure' (1 if the previous ball in the same over was a dot, 0 otherwise), interaction terms, normalized batsman average, and pitch dummies. This proxies "mental strength" as the ability to take a wicket after pressure.
  - **Example**: For a sequence in over 18: Ball 3 (Runs_Conceded=0) → pressure=1 for Ball 4. If Is_Wicket=1 on Ball 4, it supports killer instinct.
  - **Tie to Assignment**: Phase 1 (Mental Proxy)—translates coach's intuition into code, with warning handled (no pressure carry-over across overs).
- **Model Training Progress**: text

  ```
  Iter 0, Loss: 1737.3722
  Iter 2000, Loss: 644.2471
  Iter 4000, Loss: 640.1687
  Iter 6000, Loss: 640.3293
  Iter 8000, Loss: 640.8900
  Model trained.
  ```

  - **Explanation**: The Bayesian model trained for 10,000 iterations using variational inference (approximating posteriors by maximizing ELBO). Loss decreases from ~1737 (initial poor fit) to ~640 (converged), indicating the model learned patterns well.
  - **Example**: High initial loss due to noisy data (e.g., wickets from weak batsmen); convergence shows controls (pitch, batter_avg) helped isolate pressure effect.
  - **Tie to Assignment**: Phase 2 (Bayesian Inference)—uses a GLM-like model to predict 'Is_Wicket', estimating "Pressure Effect" while accounting for confounders. (PyTorch approximates PyMC, per "DCA Way" note.)
- **Inference Results**: text

  ```
  Pressure A: -0.4413 (HDI: [-0.6375832358002662, -0.2450334033370018])
  Pressure B: 2.3412 (HDI: [2.0592461466789245, 2.623120105266571])
  ```

```
Verdict: Buy Bowler B
```

- **Explanation**:
  - For Bowler A: Mean logit effect -0.4413 (negative—pressure reduces wicket odds; odds ratio exp(-0.44) ≈ 0.64, or 36% lower). 94% HDI fully negative (no zero overlap), confirming fragility.
  - For Bowler B: Mean 2.3412 (positive—increases odds exp(2.34) ≈ 10.4x). HDI fully positive, showing killer instinct.
  - Verdict: Buy B, as B's effect > A's and HDI >0.
- **Example**: If base wicket prob is 9%, after pressure: A ~5.8% (crumbles), B ~50%+ (strikes).
- **Tie to Assignment**: Phase 3 (Verdict)—Uses HDIs from posteriors to justify buying B, addressing coach's skepticism with quantifiable evidence.

Explanation of the Provided Plot Image

The attached image ("posteriors.png") shows two density plots from the model's approximate posteriors (normal distributions via variational inference):

- **Left: Posterior for Pressure Effect A**: Centered at ~ -0.44 (mean), narrow peak (low uncertainty). Fully left of zero, indicating pressure hurts A's performance (fragility).
- **Right: Posterior for Pressure Effect B**: Centered at ~2.34, also narrow. Fully right of zero, showing pressure boosts B's wickets (killer instinct).
- **Interpretation**: X-axis is logit coefficient (positive = higher wicket prob after dot); Y-axis is density. Non-overlap with zero confirms effects are credible.

How Mental Strength is Measured from the Data Using Algorithms

Mental strength ("killer instinct") is proxied as the bowler's ability to take a wicket after bowling a dot ball in death overs (per assignment hypothesis). The algorithm quantifies this via **Bayesian logistic regression** with **variational inference** (VI), controlling for confounders. Here's how, step by step, with examples:

1. **Data Preparation (Input)**: Use ball-by-ball data (e.g., 2400 death deliveries). Key fields: 'Runs_Conceded' (for dot=0), 'Is_Wicket' (target=1/0), controls like 'Pitch_Type' (dummies), 'Batter_Avg' (normalized).
   - Example: Row: Over=18, Ball=3, Runs_Conceded=0 → Sets pressure=1 for Ball=4. If Ball=4 Is_Wicket=1, suggests strength.
2. **Feature Engineering (Proxy Mental Strength)**: Create 'pressure' flag. Model only within-over (no carry-over, per warning).
   - Algorithm: Group by Match_ID, Bowler, Over; shift 'Runs_Conceded' by 1; pressure = (shifted == 0).
   - Example: For Bowler B in over 19: Dot on Ball 2 → Pressure on Ball 3. If wicket on Ball 3, boosts B's coefficient.

3. **Model Setup (Bayesian Logistic Regression)**: Predict P(Is_Wicket=1) = sigmoid(β0 + β1_pressure + β2_bowler_B + β3_pressure_bowler_B + controls).
    - Priors: β ~ Normal(0,10) (weakly informative for cricket logits ~ -3 to 3).
    - Pressure Effect: β1 (for A), β1+β3 (for B).
    - Example: If β1 = -0.44, pressure lowers A's logit by 0.44 (odds drop); β1+β3=2.34 raises B's.
4. **Training with Variational Inference (Algorithm)**: Approximate posteriors by optimizing ELBO (evidence lower bound) with Adam (lr=0.005, 10k iters, 50 MC samples). Mean-field VI assumes normal posteriors for βs.
    - Why VI? Scalable alternative to MCMC (PyMC in assignment); handles noisy data (e.g., wickets from poor pitches).
    - Example: Loss convergence (1737 → 640) shows fit; posteriors: A's mean/std → Negative effect, B's → Positive.
5. **Inference and Measurement (Output HDIs)**: Extract means/stdevs from approximated normals; compute 94% HDI (mean ± 1.88*std for normal).
    - Strength Measure: Positive effect = High strength (e.g., B's 2.34: Strong); Negative = Low (A's -0.44: Fragile). HDI non-zero confirms credibility.
    - Example: B's HDI [2.06, 2.62] >0 → 94% credible that pressure helps B (killer instinct); A's negative → Pressure hurts A.

This data-driven approach "proves" mental strength exists (contra scouts' economy focus), justifying B as the buy. The plot/HDIs provide the "graph" to convince owners. For full code rerun, fix the path as guided.