Mathematical Workflow of the ML Prefetcher

1 What the Model is Solving

At every L2C access t, you build a small feature vector $x_t \in \mathbb{R}^d$ (bias, PC bucket, page-offset bucket, last hit/miss, access type, tiny stride memory). You have a small set of actions $\mathcal{A}$ = candidate **strides** in cache lines (e.g., $\{\pm1, \pm2, \pm4, \pm8\}$).

For each action $a \in \mathcal{A}$ you keep weights $w_a \in \mathbb{R}^d$ and compute a usefulness probability

$$p_t(a) = \sigma(w_a^\top x_t) = \frac{1}{1 + e^{-w_a^\top x_t}},$$

interpreted as the probability a prefetch with stride a will be *useful* later.

This is an **online, multi-class classification** with **delayed feedback** (a contextual bandit problem).

2 Why Logistic Regression (Derivation)

We want $p(y = 1 \mid x, a)$, the probability a prefetch is useful.

For label $y \in \{0, 1\}$:

$$\Pr(y = 1 \mid x, a) = \sigma(w_a^\top x).$$

Negative log-likelihood with L2 penalty:

$$\mathcal{L}(w_a) = -\Big[y \log \sigma(z) + (1 - y) \log(1 - \sigma(z))\Big] + \frac{\lambda}{2}\|w_a\|^2, \quad z = w_a^\top x.$$

Gradient:

$$\nabla_{w_a} \mathcal{L} = (\hat{y} - y)x + \lambda w_a.$$

Update with learning rate η:

$$w_a \leftarrow (1 - \eta\lambda)w_a + \eta(y - \hat{y})x.$$

3 Decision Rule: Threshold `thr`

Each prefetch has benefit B (cycles saved) and cost C (bandwidth, pollution). Expected net gain:

$$\mathbb{E}[\Delta] = pB - (1 - p)C.$$

We issue if

$$p \geq \frac{C}{B + C} \equiv \texttt{thr}.$$

4 Cap `max_out`

Limit how many prefetches are issued per access to avoid PQ/MSHR flooding:

$$\texttt{max\_out} \leq Q_{\text{free}} - m.$$

5 Timeout

A prefetch becomes useless if its age $\geq$ `TIMEOUT`:

$$\text{if age} \geq \texttt{TIMEOUT} \quad \Rightarrow \quad y = 0.$$

6 Weight Decay λ

Weights decay geometrically:

$$\|w\| \approx (1 - \eta\lambda)^k \|w_0\| \approx e^{-\eta\lambda k} \|w_0\|.$$

Half-life:

$$k_{1/2} = \frac{\ln 2}{\eta\lambda}.$$

7 Learning Rate η

For logistic regression stability:

$$\eta \lesssim \frac{1}{\frac{1}{4}\|x\|^2 + \lambda}.$$

8 ε-Greedy Exploration

With probability ε_t, try an alternate stride:

$$\varepsilon_t = \varepsilon_{\text{start}} + (\varepsilon_{\text{end}} - \varepsilon_{\text{start}}) \min\left(1, \tfrac{t}{T_{\text{decay}}}\right).$$

9 Windowed Controller

Every W accesses:

$$\alpha = \frac{U}{\max(1, I)}, \quad \kappa = \frac{U}{\max(1, M)},$$

where I = issued, U = useful, M = demand misses.

Rules:

- If $\kappa < \kappa_{\text{min}}$ and $\alpha \geq \alpha_{\text{hi}}$: lower `thr`, raise `max_out`.
- If $\alpha < \alpha_{\text{lo}}$: raise `thr`, set `max_out`=1.

10 Full Workflow

1. Initialize: $w_a = 0$, knobs set, pending list empty.
2. On each access:
   - Build x_t, compute $p_t(a)$.
   - Apply ε-greedy.
   - Issue prefetches if $p_t(a) \geq$ `thr` up to `max_out`.
3. Track pending prefetches; mark useful ($y = 1$) or timeout ($y = 0$).
4. Update weights via SGD.
5. Every W accesses: recompute α, κ, adapt knobs.
6. Report IPC, accuracy, coverage.

11 Why It Works

- Logistic regression gives calibrated probabilities.
- Threshold balances benefit vs cost.
- Timeout enforces timeliness.
- Decay forgets stale phases.
- Exploration prevents getting stuck.
- Windowed control stabilizes accuracy and coverage.