Recommended Models

### **2. River: Adaptive Models**

[River](https://riverml.xyz/) is the best library for real-time online ML — especially in resource-sensitive environments like Kubernetes.

#### Suitable Models from River:

|  |  |  |
| --- | --- | --- |
| **Model** | **Purpose** | **Why it's great** |
| HoeffdingTreeRegressor | Adaptive trees | Captures nonlinear trends incrementally |
| AdaptiveRandomForestRegressor | Powerful ensemble | Great for mixed trends or variable load |
| ARIMA (time series) | Forecasting usage | For periodic trends (day/night patterns) |

Recommended as next-generation backend for TrendLearner

**2. HistGradientBoostingRegressor (offline batch)**

Use when:

* We’re okay with batch retraining (e.g., daily)
* We need nonlinear modeling
* Online updates are not critical per second

Good for offline retraining; not ideal for real-time Kubernetes scenarios.

**3. Vowpal Wabbit (VW)**

**Use when:**

* You have very high throughput (millions of requests per second)
* Need CLI-based stream ingestion or online logistic regression

Requires more setup, not Python-native, but extremely fast

### Best Fit for Kubernetes Resource Tuning

|  |  |  |
| --- | --- | --- |
| **Layer** | **Recommendation** | **Notes** |
| Online Trend Prediction | River.LinearRegression or HoeffdingTreeRegressor | Lightweight and adaptive |
| Ensemble Learning | AdaptiveRandomForestRegressor (River) | If usage trends vary a lot |
| Baseline | SGDRegressor (scikit-learn) | Already in use, lightweight |
| Offline Fine-tuning | HistGradientBoostingRegressor | For periodic batch updates |
| Time Series Forecasting | River.ARIMA, Prophet | If we have seasonality (e.g., diurnal patterns) |

### Suggestion for Our Project

We could extend our current TrendLearner like this:

* Keep SGDRegressor as the base for EMA + linear adaptation
* Add an optional backend switch to use River models (LinearRegression, AdaptiveRandomForest)
* Tune and evaluate using Optuna or River's Rolling evaluation
* Eventually consider serving a River model via FastAPI (using in-memory state)