Influencer Campaign Optimization

1. Sets and Parameters

- I = set of influencer records, each containing personal data (e.g., profile, audience metrics, past content types, price per content type $p_{i,c}$, etc.), indexed by i.
- C = set of content/post types, indexed by c.
- E = set of events, indexed by e.
- T_e = set of feasible posting times relative to event e, indexed by t.
- Each event e has attributes date_e, location_e, genre_e, scale_e, type_e.
- B = total campaign budget.
- Databases:
  - $\mathcal{D}_I$: Influencer analytics database with audience metrics, engagement history, genre/location affinities, prior campaign performance (excluding price per content type, which is intrinsic to influencer records).
  - $\mathcal{D}_H$: Historical campaign database with prior assignments and observed outcomes.
  - $\mathcal{D}_M$: Market and social media trend database, including seasonal effects, genre popularity, regional trends.
- $f_{i,c,e,t}(\mathcal{D}_I, \mathcal{D}_H, \mathcal{D}_M)$ = predicted effectiveness metric (impressions, engagement, ROI) informed by databases.

2. Decision Variables

$$y_{i,c,e,t} \in \{0,1\}, \quad \text{assignment indicator,} \tag{1}$$

$$b_{i,c,e,t} \geq 0, \quad \text{budget allocated to tuple } (i,c,e,t). \tag{2}$$

3. Constraints

$$\text{Budget:} \quad \sum_{i,c,e,t} b_{i,c,e,t} \leq B, \tag{3}$$

$$\text{Coupling and price:} \quad b_{i,c,e,t} \leq p_{i,c} y_{i,c,e,t}, \tag{4}$$

$$\text{Frequency/coverage/diversity:} \quad y \text{ satisfy problem-specific constraints.} \tag{5}$$

4. Objective Function

Maximize total predicted effectiveness:

$$\max \sum_{i,c,e,t} f_{i,c,e,t}(\mathcal{D}_I, \mathcal{D}_H, \mathcal{D}_M)\, y_{i,c,e,t}. \tag{6}$$

Or account for uncertainty:

$$\max \sum_{i,c,e,t} (f_{i,c,e,t} - \rho\sigma_{i,c,e,t}) y_{i,c,e,t}, \tag{7}$$

where $\sigma_{i,c,e,t}$ is prediction uncertainty and $\rho \geq 0$.

Predictive Layer (Supervised Models)

The predictive model estimates $f_{i,c,e,t}$ and optional $\sigma_{i,c,e,t}$ informed by databases $\mathcal{D}_I$, $\mathcal{D}_H$, $\mathcal{D}_M$:

- Gradient Boosted Trees (XGBoost, LightGBM, CatBoost) for tabular data.
- Deep embeddings + MLP for high-dimensional categorical inputs.
- Graph Neural Networks if social connections, audience overlaps, or clusters matter.

Optimization Layer (Discrete + Budgeted Selection)

Given predictions $f_{i,c,e,t}$, solve the combinatorial problem:

$$\begin{aligned}
\max \quad & \sum_{i,c,e,t} f_{i,c,e,t}(\mathcal{D}_I, \mathcal{D}_H, \mathcal{D}_M)\, y_{i,c,e,t} && (8)\\
\text{s.t.} \quad & \sum_{i,c,e,t} b_{i,c,e,t} \leq B, && (9)\\
& b_{i,c,e,t} \leq p_{i,c} y_{i,c,e,t}, && (10)\\
& y_{i,c,e,t} \in \{0, 1\}, && (11)\\
& \text{frequency, coverage, diversity, risk constraints.} && (12)
\end{aligned}$$

Solution Methods:

- Integer Linear Programming (ILP)
- Knapsack / Multi-knapsack formulations
- Assignment + budgeted selection models
- Reinforcement Learning for sequential or adaptive decisions

5. Construction of the Predictive Function f

The predictive function $f_{i,c,e,t}$ estimates the expected performance of assigning influencer i to post type c for event e at posting time t, informed by the available databases. Formally, we decompose f into components corresponding to each domain:

$$f_{i,c,e,t}(\mathcal{D}_I, \mathcal{D}_H, \mathcal{D}_M) = g_I(i, c, e, t; \mathcal{D}_I) + g_H(i, c, e, t; \mathcal{D}_H) + g_M(i, c, e, t; \mathcal{D}_M), \tag{13}$$

where:

- $g_I(i, c, e, t; \mathcal{D}_I)$ captures influencer-specific contributions, including intrinsic price $p_{i,c}$.
- $g_H(i, c, e, t; \mathcal{D}_H)$ captures historical campaign effects.
- $g_M(i, c, e, t; \mathcal{D}_M)$ captures market-level signals.

5.1 Formal Representation

$$g_I(i, c, e, t; \mathcal{D}_I) = h_I(\mathbf{x}_i, \mathbf{v}_c, \mathbf{e}_e, t, p_{i,c}), \tag{14}$$

$$g_H(i, c, e, t; \mathcal{D}_H) = h_H(\mathbf{h}_{i,c,e,t}), \tag{15}$$

$$g_M(i, c, e, t; \mathcal{D}_M) = h_M(\mathbf{m}_{e,t}), \tag{16}$$

where $\mathbf{x}_i$ are influencer features from the influencer record (price $p_{i,c}$ is intrinsic), $\mathbf{v}_c$ are post-type embeddings, $\mathbf{e}_e$ are event attributes, t is posting time, $\mathbf{h}_{i,c,e,t}$ encodes historical interactions, and $\mathbf{m}_{e,t}$ encodes market and temporal trends.

5.2 Usage in Optimization

The resulting $f_{i,c,e,t}$ values serve as inputs to the optimization layer, providing expected effectiveness scores for budgeted selection with temporal posting optimization and influencer-specific intrinsic pricing.

6. Predictive Function Architecture for f

The predictive function $f_{i,c,e,t}$ maps influencer records, content type, event attributes, and posting time to an expected performance metric.

6.1 Hybrid Embeddings + Gradient Boosted Trees (Recommended)

- **Input:**
  - Learned embeddings for high-cardinality categorical features: influencer ID, content type, location, genre.
  - Continuous/tabular features: influencer attributes including price per content type $p_{i,c}$, event scale, time to post relative to event.
  - Aggregated signals from historical and market databases $\mathcal{D}_I, \mathcal{D}_H, \mathcal{D}_M$ (e.g., prior campaign performance, seasonal trends, engagement statistics).
- **Benefits:**
  - Efficiently handles millions of influencer-event-content combinations via embeddings.
  - Captures complex interactions between influencers, content types, and events.
  - Gradient boosted trees provide strong predictive accuracy and interpretable feature importance for tabular features.
  - Robust to moderate-sized datasets and missing values.
  - Flexible enough to incorporate historical and market signals simultaneously.
- **Drawbacks:**
  - Requires proper embedding dimensionality tuning to avoid overfitting.

This hybrid predictive model provides the inputs $f_{i,c,e,t}$ for the downstream optimization layer. By combining learned embeddings for categorical variables with gradient boosted trees for tabular features, it balances accuracy, interpretability, and robustness, ensuring high-value recommendations for any single campaign.

7. Predictive + Optimization Architecture

The overall system for influencer campaign optimization consists of two layers: a predictive layer and an optimization layer.

7.1 Predictive Layer

- **Inputs:** Influencer records (including price per content type), content type, event features (date, location, genre, scale, type), time-to-post relative to event, and summary statistics from databases $\mathcal{D}_I$, $\mathcal{D}_H$, $\mathcal{D}_M$.
- **Categorical features:** Influencer ID, content type, location, genre.
- **Numeric features:** Price per post, audience metrics, event scale, days to post, market indicators.
- **Model Architecture:**
  - **PyTorch Embeddings + MLP:** Learns latent representations of categorical features and combines them with numeric features.
  - **Gradient Boosted Trees (LightGBM/XGBoost):** Uses embeddings + numeric features to predict expected engagement, impressions, or ROI.
- **Output:** Predicted effectiveness score $f_{i,c,e,t}$ for each influencer-content-event-time tuple.

7.2 Optimization Layer

- **Inputs:** Predicted scores $f_{i,c,e,t}$ from the predictive layer, influencer prices $p_{i,c}$, and budget/coverage/frequency/diversity constraints.
- **Problem Type:** Constrained selection (Integer Linear Programming / Knapsack / Assignment model).
- **Objective:** Maximize total predicted performance while respecting constraints:

$$\max \sum_{i,c,e,t} f_{i,c,e,t}\, y_{i,c,e,t}, \quad \text{s.t. budget, coverage, frequency, and diversity constraints.} \tag{17}$$

- **Output:** Recommended assignments $y_{i,c,e,t} \in \{0,1\}$ and allocated budgets $b_{i,c,e,t}$.

7.3 Architecture Summary

Layer	Role	Key Components
Predictive	Estimate $f_{i,c,e,t}$	PyTorch embeddings + MLP, LightGBM
Optimization	Select influencer-content-time assignments	ILP / Knapsack / Assignment solver