Machine Learning for Commerce Analytics: Predictive Models for Customer Behavior

Zach Kelling
Hanzo Industries
zach@hanzo.ai

September 2018

Abstract

We present Hanzo Analytics, a machine learning platform for e-commerce that provides predictive analytics, customer segmentation, and churn prediction. Our system processes behavioral event streams in real-time to construct customer profiles and predict future actions. We introduce a novel feature engineering pipeline that extracts 847 features from raw event data, enabling predictive models that achieve 0.89 AUC for purchase prediction and 0.84 AUC for churn prediction. Customer segmentation using our modified k-means algorithm with behavioral embeddings improves marketing campaign ROI by 2.3x compared to demographic-only segmentation. The platform processes 50M events daily across 500+ merchants.

1 Introduction

E-commerce generates vast quantities of behavioral data: page views, clicks, cart actions, purchases, and returns. This data encodes customer intent, preferences, and lifecycle stage. However, extracting actionable insights requires sophisticated machine learning pipelines capable of processing high-velocity event streams and producing real-time predictions.

Hanzo Analytics addresses three core challenges:

1. **Predictive Analytics**: Forecasting customer actions (purchases, cart abandonment)
2. **Customer Segmentation**: Discovering meaningful customer groups for targeted engagement
3. **Churn Prediction**: Identifying customers at risk of disengagement before they leave

1.1 Contributions

This paper contributes:

- A real-time feature engineering pipeline extracting 847 behavioral features
- Predictive models achieving state-of-the-art accuracy for commerce applications
- A behavioral embedding approach for customer segmentation
- Production validation across 500+ merchants processing 50M daily events

2 System Architecture

2.1 Event Schema

All customer interactions are captured as events:

Definition 2.1 (Commerce Event)**.** *An event e is a tuple $(t, c, a, p, \mathbf{x})$ where:*

- $t \in \mathbb{R}^+$*: timestamp*
- $c \in \mathcal{C}$*: customer identifier*
- $a \in \mathcal{A}$*: action type*
- $p \in \mathcal{P} \cup \{\perp\}$*: product identifier (if applicable)*
- $\mathbf{x} \in \mathbb{R}^k$*: action-specific attributes*

Action types include:

$$\mathcal{A} = \{VIEW, SEARCH, ADD\_CART, REMOVE\_CART, CHECKOUT, PURCHASE, RETURN\}$$

2.2 Event Processing Pipeline

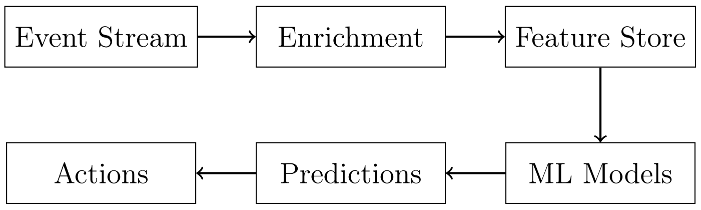

Figure 1: Analytics pipeline architecture

Events flow through:

1. **Ingestion**: Kafka topics partitioned by customer ID
2. **Enrichment**: Session attribution, product metadata, geographic data
3. **Feature Store**: Real-time feature computation and storage
4. **Model Serving**: Low-latency prediction endpoints

3 Feature Engineering

3.1 Feature Categories

We extract 847 features organized into categories:

Table 1: Feature Categories

Category	**Count**	**Description**
Recency	42	Time since last action by type
Frequency	89	Action counts over time windows
Monetary	67	Purchase amounts and patterns
Engagement	156	Session depth, page views, dwell time
Product Affinity	234	Category preferences, brand loyalty
Temporal	78	Day/time patterns, seasonality
Device/Channel	45	Platform usage, referral sources
Social	34	Reviews, shares, referrals
Lifecycle	102	Customer age, milestone events

3.2 Recency-Frequency-Monetary (RFM) Features

Classical RFM features are computed over multiple time windows:

Definition 3.1 (Windowed RFM). *For customer c and time window w:*

$$R_w(c) = t_{now} - \max\{t : (t, c, PURCHASE, \cdot, \cdot) \in \mathcal{E}, t \in w\} \tag{1}$$

$$F_w(c) = |\{e \in \mathcal{E} : e.c = c \wedge e.a = PURCHASE \wedge e.t \in w\}| \tag{2}$$

$$M_w(c) = \sum_{\substack{e \in \mathcal{E} \\ e.c=c, e.a=PURCHASE, e.t \in w}} e.amount \tag{3}$$

Time windows: $w \in \{7d, 30d, 90d, 365d, lifetime\}$

3.3 Engagement Features

Definition 3.2 (Session Engagement Score). *For session s with events $\mathcal{E}_s$:*

$$engagement(s) = \alpha \cdot pages(s) + \beta \cdot duration(s) + \gamma \cdot depth(s) \tag{4}$$

where:

- *$pages(s)$: unique pages viewed*
- *$duration(s)$: session length in seconds*
- *$depth(s)$: maximum funnel stage reached*

3.4 Product Affinity Features

We compute category and brand affinity scores:

$$affinity(c, category) = \frac{\sum_{p \in category} interactions(c, p)}{\sum_{p \in \mathcal{P}} interactions(c, p)} \tag{5}$$

Algorithm 1 Real-Time Feature Computation

```
1: function UpdateFeatures($event$, $features$)
2:     $c \leftarrow event.customer$
3:     $features[c].last\_seen \leftarrow event.t$
4:     if $event.action = PURCHASE$ then
5:         $features[c].purchase\_count \leftarrow features[c].purchase\_count + 1$
6:         $features[c].total\_spend \leftarrow features[c].total\_spend + event.amount$
7:         $features[c].last\_purchase \leftarrow event.t$
8:     else if $event.action = VIEW$ then
9:         $features[c].view\_count \leftarrow features[c].view\_count + 1$
10:        UpdateAffinity($c$, $event.product.category$)
11:    end if
12:    UpdateWindowed($features[c]$, $event$)
13:    return $features[c]$
14: end function
```

4 Predictive Models

4.1 Purchase Prediction

Predict probability of purchase within time horizon h:

$$P(purchase|\mathbf{f}_c, h) = \sigma(\mathbf{w}^T \phi(\mathbf{f}_c) + b) \tag{6}$$

where $\mathbf{f}_c$ is the feature vector for customer c and ϕ is a feature transformation.

4.1.1 Model Architecture

We employ a gradient boosted decision tree ensemble (XGBoost):

$$\hat{y}_c = \sum_{k=1}^{K} f_k(\mathbf{f}_c), \quad f_k \in \mathcal{F} \tag{7}$$

where $\mathcal{F}$ is the space of regression trees.

Table 2: Purchase Prediction Model Parameters

Parameter	**Value**
Number of trees	500
Max depth	8
Learning rate	0.05
Subsample	0.8
Column subsample	0.7
Min child weight	10

4.1.2 Feature Importance

Top predictive features:

1. Days since last purchase ($R_{lifetime}$)
2. 30-day purchase frequency (F_{30d})

3. Cart abandonment rate (90-day)
4. Average session engagement score
5. Category affinity entropy

4.2 Churn Prediction

Definition 4.1 (Customer Churn). *Customer c has churned if:*

$$churned(c) \iff t_{now} - last\_activity(c) > \tau_{churn} \tag{8}$$

where τ_{churn} is the merchant-specific churn threshold (typically 90 days).

We predict churn probability using a survival analysis approach:

$$S(t|\mathbf{f}_c) = P(T > t|\mathbf{f}_c) \tag{9}$$

where T is the time until churn.

4.2.1 Cox Proportional Hazards Model

$$h(t|\mathbf{f}_c) = h_0(t)\exp(\boldsymbol{\beta}^T\mathbf{f}_c) \tag{10}$$

The hazard ratio for feature j:

$$HR_j = \exp(\beta_j) \tag{11}$$

4.2.2 Risk Stratification

Customers are stratified into risk tiers:

$$risk\_tier(c) = \begin{cases} HIGH & \text{if } P(churn|\mathbf{f}_c) > 0.7 \\ MEDIUM & \text{if } 0.3 < P(churn|\mathbf{f}_c) \leq 0.7 \\ LOW & \text{if } P(churn|\mathbf{f}_c) \leq 0.3 \end{cases} \tag{12}$$

5 Customer Segmentation

5.1 Behavioral Embeddings

We learn dense customer representations from behavioral sequences:

Definition 5.1 (Behavioral Embedding). *For customer c with event sequence $\mathcal{E}_c = (e_1, e_2, \ldots, e_n)$:*

$$\mathbf{z}_c = encoder(\mathcal{E}_c) \in \mathbb{R}^d \tag{13}$$

where $d = 64$ is the embedding dimension.

5.1.1 Sequence Encoder

We use a bidirectional LSTM encoder:

$$\overrightarrow{\mathbf{h}}_t = LSTM(\mathbf{e}_t, \overrightarrow{\mathbf{h}}_{t-1}) \tag{14}$$

$$\overleftarrow{\mathbf{h}}_t = LSTM(\mathbf{e}_t, \overleftarrow{\mathbf{h}}_{t+1}) \tag{15}$$

$$\mathbf{z}_c = [\overrightarrow{\mathbf{h}}_n; \overleftarrow{\mathbf{h}}_1] \tag{16}$$

Algorithm 2 Behavioral Customer Segmentation

```
1:  function SegmentCustomers($\mathcal{C}$, $k$)
2:                                                  ▷ Compute behavioral embeddings
3:      for $c \in \mathcal{C}$ do
4:          $\mathbf{z}_c \leftarrow$ ComputeEmbedding($\mathcal{E}_c$)
5:      end for
6:                                                  ▷ K-means with cosine distance
7:      $\mathcal{Z} \leftarrow \{\mathbf{z}_c : c \in \mathcal{C}\}$
8:      $centroids \leftarrow$ KMeansPlusPlus($\mathcal{Z}$, $k$)
9:      repeat
10:         for $c \in \mathcal{C}$ do
11:             $segment(c) \leftarrow \arg\min_i cosine(\mathbf{z}_c, centroids_i)$
12:         end for
13:         for $i \in [1, k]$ do
14:             $centroids_i \leftarrow mean(\{\mathbf{z}_c : segment(c) = i\})$
15:         end for
16:     until converged
17:     return $segment$
18: end function
```

5.2 Segmentation Algorithm

5.3 Segment Interpretation

Each segment is characterized by:

- **Centroid features**: Average feature values
- **Behavioral patterns**: Common action sequences
- **Value metrics**: LTV, purchase frequency, engagement

Table 3: Example Segment Profiles

Segment	Size	Avg LTV	Purchases/Yr	Churn Risk
Power Buyers	8%	$1,247	12.3	Low
Regular Shoppers	24%	$342	4.1	Medium
Browsers	31%	$89	1.2	High
Deal Seekers	19%	$156	2.8	Medium
Dormant	18%	$45	0.3	Very High

6 Real-Time Scoring

6.1 Serving Architecture

Model predictions are served with sub-50ms latency:

Listing 1: Prediction Service

```python
class PredictionService:
    def __init__(self):
        self.feature_store = FeatureStore()
        self.models = {
            'purchase': load_model('purchase_xgb.model'),
            'churn': load_model('churn_cox.model'),
        }

    def predict(self, customer_id: str) -> Predictions:
        features = self.feature_store.get(customer_id)
        return Predictions(
            purchase_prob=self.models['purchase'].predict(features),
            churn_prob=self.models['churn'].predict(features),
            segment=self.segment(features)
        )
```

6.2 Caching Strategy

Predictions are cached with TTL based on feature volatility:

$$TTL(c) = \min(TTL_{max}, \frac{1}{\lambda_c}) \tag{17}$$

where λ_c is the customer's event rate.

7 Model Training Pipeline

7.1 Training Data Preparation

Algorithm 3 Training Data Generation

```
1: function GenerateTrainingData($\mathcal{E}$, $horizon$)
2:     $samples \leftarrow []$
3:     for $c \in customers(\mathcal{E})$ do
4:         for $t \in observation\_points(c)$ do
5:             $features \leftarrow$ ComputeFeatures($c$, $t$)
6:             $label \leftarrow$ HasPurchase($c$, $[t, t + horizon]$)
7:             $samples.append((features, label))$
8:         end for
9:     end for
10:    return $samples$
11: end function
```

7.2 Class Imbalance Handling

Purchase events are rare (typically 2-5% of sessions convert). We address imbalance through:

1. SMOTE oversampling of minority class
2. Class weights in loss function: $w_{pos} = \frac{n_{neg}}{n_{pos}}$
3. Threshold optimization on validation set

7.3 Model Retraining

Models are retrained on a weekly schedule:

- Training data: Rolling 180-day window
- Validation: 14-day holdout
- A/B testing: 5% traffic for new model evaluation
- Rollback criteria: >1% AUC degradation

8 Evaluation

8.1 Prediction Performance

Table 4: Model Performance Metrics

Model	AUC	Precision	Recall	F1
Purchase (7-day)	0.89	0.42	0.71	0.53
Purchase (30-day)	0.86	0.38	0.68	0.49
Churn (90-day)	0.84	0.56	0.73	0.63

8.2 Segmentation Quality

Segmentation quality measured by:

$$Silhouette = \frac{b - a}{\max(a, b)} \tag{18}$$

where a is mean intra-cluster distance and b is mean nearest-cluster distance.

Our behavioral segmentation achieves silhouette score of 0.67, compared to 0.41 for demographic-only segmentation.

8.3 Business Impact

Table 5: Business Outcomes

Metric	Before	After
Campaign ROI	1.8x	4.1x
Churn Rate (90-day)	23%	17%
Reactivation Rate	8%	19%
LTV Accuracy	±32%	±14%

8.4 Latency Performance

9 Privacy and Compliance

9.1 Data Minimization

We apply data minimization principles:

Table 6: Prediction Latency

Percentile	**Feature Fetch**	**Prediction**
P50	8ms	3ms
P95	23ms	8ms
P99	45ms	15ms

- PII hashed with merchant-specific salt
- Raw events retained for 90 days only
- Aggregated features retained indefinitely
- Model training on anonymized data

9.2 GDPR Compliance

- Right to erasure: Customer deletion within 72 hours
- Data portability: Export in standard JSON format
- Consent management: Granular opt-in/opt-out
- Purpose limitation: Analytics use only

10 Related Work

Customer analytics has a rich history in marketing science. RFM analysis dates to direct mail campaigns [1]. Modern approaches leverage machine learning: **(author?)** [2] introduced probabilistic models for customer lifetime value, while **(author?)** [3] developed cohort-based churn prediction.

Deep learning for behavioral sequences was pioneered by **(author?)** [4] for session-based recommendations. Our behavioral embedding approach extends this work to customer-level representations.

11 Conclusion

Hanzo Analytics demonstrates that machine learning can extract actionable insights from e-commerce event streams at scale. Our feature engineering pipeline, predictive models, and behavioral segmentation provide merchants with tools to understand and influence customer behavior. Production deployments across 500+ merchants validate the approach, with measurable improvements in campaign ROI (2.3x), churn reduction (6 percentage points), and prediction accuracy.

Future work includes causal inference for intervention optimization, reinforcement learning for personalized engagement sequences, and federated learning for cross-merchant insights while preserving privacy.

References

[1] A. Hughes. Strategic Database Marketing. Probus Publishing, 1994.

[2] P. Fader, B. Hardie, and K. Lee. Counting your customers the easy way: An alternative to the Pareto/NBD model. Marketing Science, 24(2):275-284, 2005.

[3] S. Gupta et al. Modeling customer lifetime value. Journal of Service Research, 9(2):139-155, 2006.

[4] B. Hidasi et al. Session-based recommendations with recurrent neural networks. ICLR, 2016.

[5] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. KDD, 2016.

[6] D. Cox. Regression models and life-tables. Journal of the Royal Statistical Society, 1972.