Peer-Based Feature Construction Using Learned Embeddings

This document explains how peer-based features are added to a panel dataset with a MultiIndex of `['date', 'permno']`. The goal is to introduce a cross-sectional "peer effect" by aggregating information from firms that are similar in a **learned embedding space**. The embeddings come from a neural network trained to predict next-period returns. Firms with similar embeddings are assumed to share related economic characteristics, and therefore can be treated as meaningful peers.

This procedure is performed **date-by-date** to avoid any temporal leakage and to ensure that peers are defined only within the same cross-section.

Overview of the Approach

For each date t and firm i:

1. Extract the firm's embedding vector $z_{i,t}$, produced by a neural network trained on information available at time t.
2. Compute distances between $z_{i,t}$ and other firms' embeddings $z_{j,t}$ in that date's cross-section.
3. Convert these distances into **similarity weights**. The preferred method uses a **Gaussian similarity kernel**.
4. Normalize the weights so that they sum to 1 across all peers.
5. Form a **peer-weighted feature** by taking a weighted average of some firm-level variable $q_{j,t}$ (e.g., predicted forward return, lagged return, or other characteristics).
6. Assign the resulting peer feature back to the corresponding firm–date pair.

This produces a new variable:

$$\text{peer\_feat}_{i,t} \;=\; \sum_{j \neq i} w_{ij,t}\, q_{j,t}$$

where $w_{ij,t}$ reflects how similar firms j are to firm i on date t.

The entire process is repeated independently for each date.

Why Use Embedding-Based Peers?

The embedding learned by the neural network represents a compressed summary of the economic state of each firm. Using distances in embedding space allows peers to be defined *endogenously* based on return-relevant information, rather than relying on broad categories such as industries.

This makes peer groups more granular and adaptive across time.

Step-by-Step Construction

Date-by-Date Cross-Section

Peer calculations are performed separately for each date. For each date t:

- Extract embedding vectors $\{z_{i,t}\}_{i=1}^{N_t}$.
- These vectors represent all firms active at time t.
- Distances and weights are computed only within this cross-section.

This ensures that all information used to construct the peer feature is contemporaneously available and avoids look-ahead bias.

Computing Peer Similarities

For each firm i at time t, compute distances to all other firms:

$$d_{ij,t} = \|z_{i,t} - z_{j,t}\|_2.$$

These distances quantify how close firm j's embedding is to firm i's embedding. Smaller distances imply greater similarity.

Gaussian Similarity Weights

Distances are converted into **Gaussian kernel** similarity weights:

$$w_{ij,t} \;=\; \exp\!\left(-\frac{d_{ij,t}^2}{2\,\mathrm{bw}_t^2}\right),$$

where bw_t is the bandwidth, a scaling parameter controlling how rapidly similarity declines with distance.

Key properties of Gaussian similarity:

- Weights are strictly between 0 and 1.
- Very close peers receive weights near 1.
- Distant firms receive weights near 0.
- Similarity decays smoothly and rapidly with distance.
- No division-by-zero issues.

The diagonal weight $w_{ii,t}$ is set to zero to avoid assigning self-weight.

Finally, weights are normalized:

$$\tilde{w}_{ij,t} = \frac{w_{ij,t}}{\sum_{k \neq i} w_{ik,t}}$$

so that they sum to 1 across all peers of firm i.

Choosing a Bandwidth

The bandwidth determines how "local" the peer group is:

- **Small bandwidth**: only very close embeddings contribute meaningfully.
- **Large bandwidth**: many firms contribute moderately.

If not explicitly set, a common approach is:

- bandwidth = median of non-zero pairwise distances for that date.

This adapts the kernel to the scale of the data at each cross-section.

Aggregating Peer Information

Once weights are computed, form the peer feature using any contemporaneously available variable $q_{j,t}$:

$$\text{peer\_feat}_{i,t} \;=\; \sum_{j \neq i} \tilde{w}_{ij,t}\, q_{j,t}.$$

Examples of $q_{j,t}$:

- Predicted next-period return from the neural network (safe: uses information at time t)
- Lagged realized return (e.g., $r_{j,t}$)
- Firm characteristics such as size, volatility, or valuation ratios

Importantly, $q_{j,t}$ must use information known at date t — never future returns.

Resulting Peer Feature

The resulting variable:

- is firm-specific and time-specific,
- reflects influence from similar firms,
- is grounded in a learned similarity measure,
- and can be used as an input to subsequent predictive models or factor analyses.

If a cross-section has no valid peers (e.g., only one active firm), the feature is assigned `NaN`.

Comparison: Gaussian Similarity vs Inverse-Distance Weighting

Two standard methods exist for turning distances into weights. Here we compare them conceptually.

Gaussian Similarity Weights

$$w_{ij} = \exp\left(-\frac{d_{ij}^2}{2\,\mathrm{bw}^2}\right)$$

Advantages:

- Smooth, stable weighting.
- Strong emphasis on close neighbors; distant neighbors effectively ignored.
- Naturally bounded between 0 and 1.
- No singularity at zero distance.
- Produces controlled, symmetric influence patterns.
- Easy to adapt across time via bandwidth selection.

Disadvantages:

- Requires choosing a bandwidth parameter.
- The choice of bandwidth can influence locality of peer behavior.

Inverse-Distance Weights

$$w_{ij} = \frac{1}{d_{ij} + \epsilon}$$

Advantages:

- Simple to understand.
- Close peers receive high weights.

Disadvantages:

- Singular at $d_{ij} = 0$; needs a stability constant ϵ.
- Weight decays too slowly for far-away firms, causing broad averaging.
- More sensitive to noisy distances.
- Less controlled locality compared to Gaussian kernel.

Summary of Comparison

Criterion	Gaussian Similarity	Inverse Distance
Smoothness	Very smooth	Less smooth
Decay rate	Fast, tunable via `bw`	Slow
Numerical stability	Excellent	Requires ϵ
Emphasizes close peers	Yes	Yes
Ignores far peers	Effectively yes	Not strongly

Gaussian similarity is preferred for stability, interpretability, and better control of local peer effects.

Final Notes

The peer-based feature constructed through Gaussian similarity weighting provides a richer, more adaptive measure of cross-sectional influence. When combined with embeddings learned from a predictive neural network, this approach captures nuanced relationships among firms that traditional industry classifications or simple distance metrics cannot.

This peer feature can then be used in downstream models such as ElasticNet, gradient boosting, or further neural networks, or evaluated directly as a return-predictive factor.