Machine Learning and Deep Learning–Enhanced Feature Engineering and Model Zoo Ensembling for the Trexquant Market Prediction Task

Kexin Deng (kd537), Michael Cao (yc849), Yichen Gao (yg635)

ABSTRACT

This project presents an integrated machine learning framework for cross-sectional market prediction in the Trexquant competition. We develop a unified pipeline that combines domain-informed feature engineering with both machine learning and deep learning components. The model architecture incorporates gradient boosting methods, regularized linear models, and an IC-oriented neural network, each trained under randomized feature subsets and multi-seed configurations to enhance diversity. A structured Model Zoo is constructed to generate complementary predictions, which are then aggregated through a constrained ensemble optimizer. This approach yields a stable and generalizable forecasting system that effectively captures nonlinear relationships, mitigates overfitting, and improves out-of-sample performance on the Trexquant dataset.

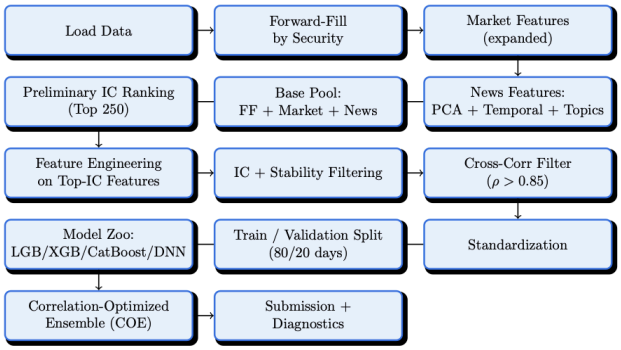

Figure 1: Overall pipeline of our strategy

DATA DESCRIPTION

The dataset provided by Trexquant consists of cross-sectional daily observations of U.S. equities, combining structured financial variables, engineered alpha signals, and news-based embedding vectors. The official assignment document already describes the raw variables in detail.

METHODOLOGY

Phase 1: Data Cleaning & Forward-fill by Security

The dataset is intrinsically panel-structured (organized by stock index *si* and day index *di*). It is a 400-dimensional alpha-signal matrix, where each column represents a distinct proprietary quantitative feature and each row corresponds to a specific stock-day observation; the signals are already normalized, but their exact definitions/names are not disclosed by the competition organizers.

Phase 2: Market Features

We construct a market-level feature set that includes 55 features (18 baseline, 37 expanded). The baseline set captures liquidity, news flow, announcement timing, short-horizon returns, SPY benchmarks, VIX levels, and sector or industry identifiers. The new feature set improves how we represent market structure by adding several important measures. It includes volume-based microstructure for liquidity, signals for multi-horizon returns that extend daily momentum, and terms that connect returns across different timeframes and benchmarks. Further improvements involve applying simple nonlinear changes to announcement timing fields, adding VIX-based sensitivity to volatility regimes through adjusted returns, and using smooth sinusoidal encodings for sector and industry identifiers. Together, these additions offer a clearer summary of current market conditions.

Phase 3: News Embeddings

We enhance each observation with signals from news by linking it to a pretrained 768-dimensional embedding. These embeddings connect to security-day identifiers and compress to 64 PCA components, offering a compact summary of news content. To capture how news changes over time, we calculate temporal transforms of the PCA vectors for each security. This includes one-day and multi-day differences, short rolling averages and standard deviations, and cosine similarities with both the previous embedding and the local window. These steps provide 258 temporal features that show changes in information rather than just static content. We also introduce a broad topical structure by applying 20-cluster K-Means to the PCA embeddings and using the resulting assignments as 20 one-hot topic indicators. Combining PCA features, temporal dynamics, and topic encodings generates 342 final news features, which represent the content, evolution, and thematic context of security-specific news.

Phase 4: Feature Pool Construction and Refinement

We start by creating a unified base feature pool of 797 signals that combines the 400 security-level alphas with engineered market and news-derived features. Then, we calculate the preliminary cross-sectional Information Coefficient (IC) for each feature against the target. We then select the top 250 features based on their absolute IC. These high-quality signals form the basis for targeted nonlinear expansions. For each selected feature, we create simple polynomial-style transforms, including squared, absolute-value, sign-sqrt forms, and clipped variants that reduce extreme sensitivities. We also build interaction terms by combining the top features with key market variables like recent returns, SPY-relative performance, and VIX. This captures risk-dependent nonlinearities and increases the total to 1,947 features before filtering.

To narrow down this expanded pool to a stable set of predictors, we follow a two-stage filtering process. First, we use an IC-stability screen to evaluate each feature's global IC and the consistency of its sign across various time splits. We retain features that are both strong and temporally stable, resulting in 600 candidates. Next, we apply a cross-correlation filter that groups highly collinear features, keeping only the feature within each group that has the strongest correlation to the target. This step reduces redundancy and gives us a compact set of 237 decorrelated signals.

Finally, we standardize the retained features to ensure they are comparable across scales, preparing the complete modeling feature

set for all subsequent training. For model evaluation, we divide the training period by time. The last 20% of trading days go into a validation set. This split keeps the time sequence intact so we can evaluate how well the model performs out-of-sample.

Phase 5: Model Zoo Construction and Ensemble

We employed random feature bagging, where each model instance receives only a subset of the full feature set, to increase model diversity, reduce multicollinearity, and decrease variance while improving generalization. We also trained models across multiple random seeds for more stable and robust performance. Downside is that seed sensitivity and randomness is inherent. Still it's the best ensemble we had across other alternatives.. Ultimately, we selected the Correlation-Optimized Ensemble (COE)-based ensemble with strict weight control as our final approach. It selects model weights by minimizing prediction error under the constraint of keeping models low-correlated, so that the final ensemble maximizes stability, diversity, and out-of-sample performance.

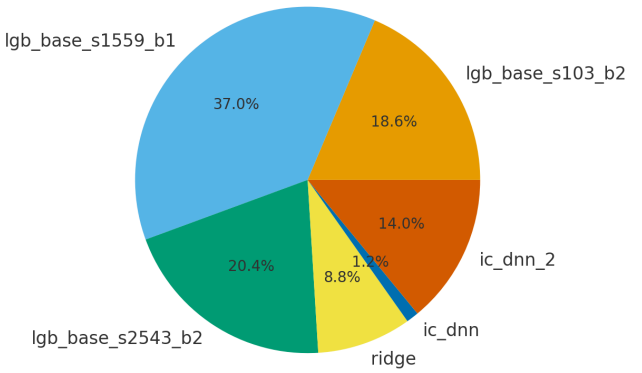

Figure 2: Model Weights and Components

Our final submission ensemble places most of weight on 3 LightGBM models: lgb_base_s1559_b1 (130 features), lgb_base_s2543_b2 (165 features), lgb_base_s103_b2 (165 features), as they consistently demonstrated the strongest IC performance and the highest overall stability. The two IC-DNN models provided additional diversity by capturing nonlinear patterns through their IC-driven training objective. Finally, the Ridge model offered a simple but highly stabilizing linear component, improving robustness and helping reduce overfitting in the overall ensemble.

We acknowledge that XGBoost performed very well in several earlier submissions, sometimes even outperforming LightGBM; however, in this final version it did not receive meaningful ensemble weight, probably due to some parameter setting.

MODELS ATTEMPTED BUT FAILED

We believe it is also meaningful to briefly document the models that did not work as expected. Exploring why certain approaches failed helped us better understand the data and refine our pipeline. They are the pathways that led to our final model.

The number of features used in our final models: a wide range of configurations 60–80, ~120, 200, 200+, 400, 500, 800, and 1200. Our final version was around 2000 and then downsized.

Automated feature generation packages: AutoFeat (noisy)

Feature manipulation methods: additional experiments with automated feature selection tools inspired by FeatureWiz, including: Mutual information ranking, SULOV/ MRMR to assess mutual information and pairwise correlations for a more compact and stable core feature subset.

We also tried latent factor methods. PCA worked for news part but not for Namely, PCA for extracting principal linear components; ICA for identifying statistically independent factors; SGM (Signal Grouping Method) to cluster correlated alphas and summarize them into group-level factors; Sparse Autoencoder (SAE) to construct nonlinear compressed factors with sparsity constraints. These lower-dimensional latent representations were before incorporated alongside baseline features to enrich model expressiveness, but then took out because of poor performance.

More machine learning and deep learning models: ExtraTrees, Random Forest (too slow), deep learning models like MLP (dead kernel), CatBoost (quick but not as good), Lasso (too low and unstable), TabNet (slow, fragile and bad)

Tuning methods: grid search (too slow), OPTUNA (too time consuming for so many models we are implementing. We did a version for once or twice, but feature also changes)

Ensemble Methods: Finally we chose COE (but with weight control), but we passed NNLS, correlation-weighted rank average, PCA-decorrelated softmax Ridge ensemble.

RESULTS

Throughout development, we maintained multiple versions of our pipeline, testing and isolating each block individually. Our internal evaluation was based on Pearson correlation computed on our own validation split, which naturally introduced differences between our estimates and the leaderboard results. Performance also varied across random seeds, adding another layer of variability to each model version.

In terms of effort, the project grew significantly: the code and data on Kexin's laptop reached 155.46 GB, and through extensive optimization we reduced the runtime from over 7.5 hours per run to a more manageable 2 hours.

Over the two-month competition period, we submitted 29 versions of our model in total. Only resting a few days when we try to perfect our models. Our best public score reached 0.07066, and our best private score reached 0.07055. The submission we ultimately selected ranked Top 2 on the Public Leaderboard and Top 3 on the Private Leaderboard.

ACKNOWLEDGEMENT

We would like to thank Trexquant and everyone who made this competition possible. The high-quality data and alpha features provided gave us a solid foundation so that it's such a great opportunity even if we don't win anything. We'd love to thank our computers. We named the team because by the last day, we joked that we wish to have a ThinkSystemSR780aV3 model of NVIDIA GPU since we can only rely on our own poor laptops which broke down with dead kernels and memories nearly every day.

We're also grateful to those teams who worked hard alongside us. We encouraged each other, celebrated when they climbed up. Some of their members put in so much effort and carried the team even until the final day and final hour. We were happy even if they exceeded us. And lastly, we want to thank each other. It was an equal contribution and definitely one of the teamwork experiences we've had. The three of us collaborated on ORIE 5259 and got to do one more project together before graduation. We kept each other's company until the very end.