LLMs on Tabular Data with Limited Semantics: Evidence from Industrial Car Retrofit Prediction

Anonymous ACL submission

Abstract

Industrial retrofit planning depends on structured operational data rather than free text: planners must estimate whether a newly registered prototype will require a retrofit, which retrofit package it will need, and how long the work will take. We study an industrial dataset linking a prototype-registration system (284,271 vehicles) with a retrofit-management system (48,716 cleaned visits), and compare strong tabular machine learning baselines with three LLM-based strategies on row-serialized inputs: embedding features, direct prompted classification, and an ML+LLM stacking approach. Across binary occurrence prediction, 15-way retrofit-type classification, per-visit duration regression, and an aggregated monthly benchmark, classical tree ensembles remain the strongest standalone models. However, the LLM results reveal a consistent pattern: embeddings remain useful on tables (binary AUC $= 0.982$), direct prompting collapses once semantic signal is stripped by hashing (binary AUC $= 0.500$; multiclass weighted F1 $= 0.018$), and hybrid stacking yields the best manually built multiclass model (weighted F1 $= 0.626$). The results suggest that on privacy-constrained industrial tables, LLMs are more effective as complementary components than as replacements for strong tabular baselines.

1 Introduction

Retrofit departments in automotive development modify prototype vehicles by implementing hardware and software changes so that downstream testing can proceed. Capacity planning is difficult because early vehicle metadata is high-dimensional, mostly categorical, and noisy, while the retrofit demand signal is rare. In practice, planners need answers to three questions: *will* a newly registered prototype visit the retrofit department, *what* retrofit package will it need, and *how long* will the work take?

We study this pipeline through the lens of LLMs over structured data. The inputs are enterprise tables whose categorical values are hashed, so row serialization exposes structure but little lexical semantics. This makes the setting useful for separating what LLM-based methods can learn from distributional patterns alone from what they need semantic content to solve. We compare strong tabular baselines against three LLM-based strategies on serialized rows, and also benchmark time-series foundation models on an aggregated monthly planning signal.

Contributions. We make three contributions. First, we formulate an end-to-end industrial planning problem over structured data: binary occurrence prediction, 15-way retrofit-type prediction, per-visit duration regression, and monthly forecasting. Second, we compare embedding features, prompted classification, and ML+LLM stacking against strong classical baselines and an AutoML reference. Third, we report a practical failure mode: direct prompting performs poorly when categorical semantics are hidden, while embeddings and hybrid stacking still retain useful signal.

2 Related work

Recent work studies whether language-model interfaces can improve prediction on tables by serializing rows as text, learning embedding features, or prompting models directly (Dinh et al., 2022; Hegselmann et al., 2023). Tabular foundation models such as TabPFN ask a related question for pretrained transformers over structured inputs (Hollmann et al., 2023). For high-cardinality industrial tables, however, target encoding (Micci-Barreca, 2001), boosted trees such as XGBoost, LightGBM, and CatBoost (Chen and Guestrin, 2016; Ke et al., 2017; Prokhorenkova et al., 2018), and imbalance-aware evaluation remain strong baselines (Saito and Rehmsmeier, 2015; Chawla et al., 2002). For

the temporal side, we compare classical statistical forecasting (Box and Jenkins, 1990; Hyndman and Athanasopoulos, 2021; Taylor and Letham, 2018), lag-based machine learning, and recent time-series foundation models including Chronos and TIME-LLM (Ansari et al., 2024; Jin et al., 2024).

3 Data and tasks

We use two internal systems from a large automotive manufacturer: a *prototype-registration system* with early vehicle metadata, and a *retrofit-management system* with retrofit package labels and process timestamps. All categorical values are hashed; continuous features are engineered from timestamps, fleet counts, and frequency statistics. For LLM-based methods, each row is also serialized into a key–value description that can be embedded or passed to a prompted classifier.

A left join produces the binary occurrence dataset (n=284,271), where the target indicates whether the vehicle ever visits the retrofit department (positive rate about 4.4%). An inner join produces the retrofit-only dataset (n=54,174 visits before duration filtering), used for retrofit-type prediction and duration regression. After grouping rare multi-type combinations into their rarest component, the retrofit-type task contains 15 classes. For duration, we remove P1–P99 outliers, leaving 52,507 visits; the right-skewed target is modeled with a $\log(1+x)$ transform. Finally, we construct a 76-point monthly time series of mean retrofit duration as a low-data planning benchmark.

4 Methods and experimental setup

Classical baselines. Numeric features include vehicle age, fleet-size counts aggregated by derivative, base type, engine type, and build location, as well as interactions such as age$\times$fleet and priority$\times$fleet. To mitigate leakage, we exclude direct identifiers and temporal fields that would trivially reveal the target. After target encoding, frequency encoding, and importance-based filtering, the binary task retains 44 features and the multiclass/temporal tasks retain 50 features each. We evaluate logistic regression, Random Forest (Breiman, 2001), Extra Trees, Gradient Boosting, XGBoost, LightGBM, CatBoost, stacking, and voting ensembles. Hyperparameters are tuned with Optuna (Akiba et al., 2019); for multiclass prediction, the SMOTE ratio is tuned jointly with model hyperparameters.

LLM-based strategies. LLM-EMBED serializes each row and embeds it with Amazon Titan Embed v2; a logistic regression classifier is trained on the resulting vectors. Because of API cost, embedding experiments use 5,000 training rows. LLM-PROMPTED gives Claude Sonnet 4 each serialized row with task-specific few-shot examples and asks for the label directly; prompted inference is evaluated on smaller held-out subsets. LLM-STACKED combines the best classical probabilities with LLM-derived outputs in a meta-learner, testing whether LLMs are more useful as components than as standalone predictors. AutoGluon provides an AutoML reference (Erickson et al., 2020). For the monthly benchmark, we compare statistical baselines, ML on lag features, AutoGluon on lag features, Chronos, and TIME-LLM; related foundation-model alternatives include TimeGPT, Lag-Llama, and decoder-only time-series transformers (Garza et al., 2024; Rasul et al., 2024; Das et al., 2024).

Evaluation. We use an 80/20 stratified train–test split for classification and regression, and a 16-month holdout for the monthly time series. Metrics are ROC-AUC, PR-AUC, and thresholded F1 for Stage 1; accuracy, weighted F1, macro F1, and one-vs-rest AUC for Stage 2; and MAE, RMSE, and R^2 for Stage 3. Because Stage 1 is heavily imbalanced, ROC-AUC is interpreted jointly with PR-AUC (Saito and Rehmsmeier, 2015).

5 Results

Stage 1: occurrence prediction. Table 1 shows the main binary results. Hyperparameter-tuned CatBoost achieves the strongest standalone performance (AUC $= 0.997$, F1 $= 0.884$), while the voting ensemble yields the best PR-AUC. Among LLM-based methods, embeddings remain useful on rows (AUC $= 0.982$), but direct prompting collapses to random performance (AUC $= 0.500$). The hybrid stack attains F1 $= 0.900$ on its smaller LLM evaluation subset, suggesting complementary signal, but strong tabular models remain the most reliable standalone predictors.

Stage 2: retrofit-type prediction. The multiclass task provides the clearest positive result for hybrid LLM use. As shown in Table 2, LLM-STACKED reaches the best weighted F1 among manually configured pipelines (0.626), slightly ahead of Random Forest with SMOTE (0.621) and

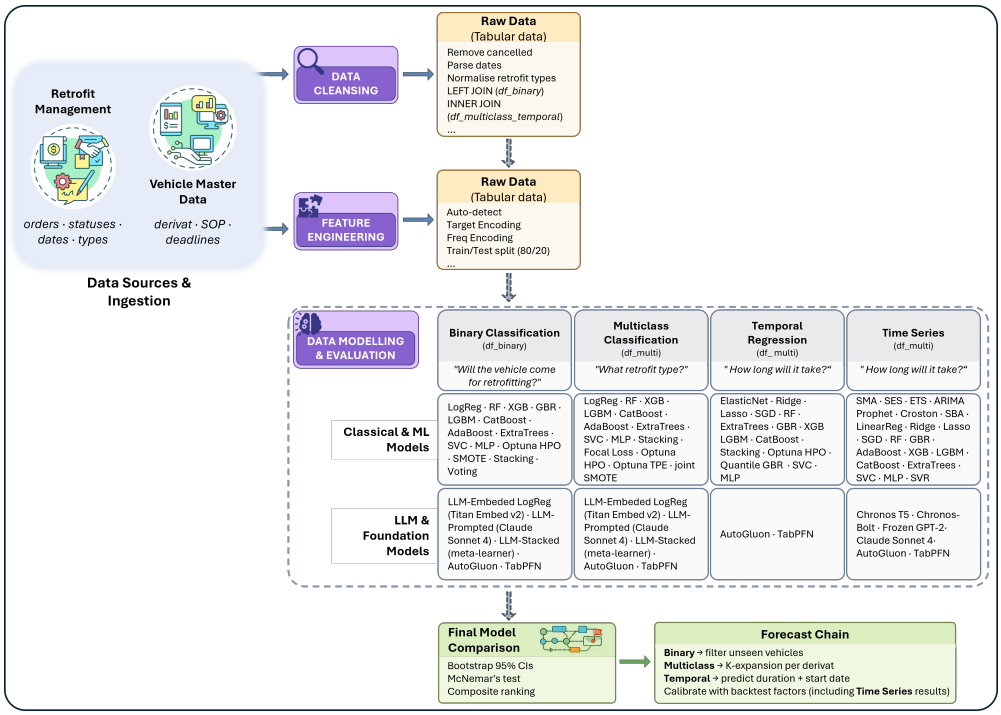

Figure 1: Modeling pipeline from data cleansing and feature engineering to binary occurrence prediction, retrofit-type classification, duration regression, monthly forecasting, and final model comparison.

Table 1: Stage 1 binary occurrence prediction. † indicates the smaller 200-row LLM evaluation subset.

Model	ROC-AUC	PR-AUC	F1
CatBoost (HP-tuned)	**0.997**	0.932	0.884
Voting ensemble	0.997	**0.937**	0.883
LLM-STACKED†	0.996	0.897	**0.900**
LLM-EMBED	0.982	0.667	0.684
LLM-PROMPTED†	0.500	0.045	0.086
AutoGluon	0.997	—	0.881

Table 2: Stage 2 retrofit-type prediction.

Model	Acc	$F1_w$	$F1_m$
AutoGluon	**0.702**	**0.654**	0.348
LLM-STACKED	0.680	0.626	0.281
Random Forest (SMOTE)	0.611	0.621	**0.390**
XGBoost (SMOTE)	0.601	0.614	0.380
LLM-EMBED	0.480	0.521	0.327
LLM-PROMPTED	0.100	0.018	0.012

XGBoost (0.614). AutoGluon remains best overall at 0.654 weighted F1. Again, direct prompted classification performs poorly on hashed inputs, while embedding-based features remain serviceable but below the top tabular baselines.

Stage 3 and monthly planning. For per-visit duration, classical ensembles still dominate: AutoGluon achieves MAE = 4.9 days and Extra Trees achieves MAE = 5.1 days. On the monthly series, lag-based ML remains strongest, but Chronos-small is competitive without task-specific training. The results suggest that for temporal planning, foundation models are promising complements but not replacements for compact supervised baselines.

Table 3: Duration and monthly benchmark results.

Setting	Model	MAE	R^2
Per-visit	AutoGluon	**4.9**	**0.38**
	Extra Trees	5.1	0.37
Monthly	LightGBM (lags)	**3.16**	-1.90
	AutoGluon (lags)	3.16	-1.88
	Chronos-small	4.03	-0.07
	TIME-LLM	4.62	-0.22

6 Discussion and deployment considerations

Why direct prompting fails here. The core pattern is consistent across tasks. Once categorical values are hashed, direct prompted classification loses the semantic cues that make language-model reasoning useful. The model sees a row-shaped string, but most values are opaque identifiers rather than meaningful descriptors. This explains why prompting collapses in both the rare-event binary task and the 15-way retrofit-type task: the prompt format provides access to feature names and examples, but not to the private domain semantics hidden behind the hashes. By contrast, row embed-

dings still preserve co-occurrence and frequency structure, and stacking can exploit complementary error patterns.

Where LLM components help. The strongest standalone systems remain classical tabular models, especially target-encoded boosted trees and AutoML ensembles. The positive LLM result is therefore not that a prompted model replaces the tabular pipeline, but that LLM-derived features can be useful when inserted at the right interface. LLM-EMBED offers a compact representation of serialized rows, and LLM-STACKED improves the manually configured multiclass pipeline by combining tabular probabilities with LLM-derived outputs. This suggests a practical design rule for privacy-constrained industrial tables: use LLMs as feature providers or auxiliary components, not as the sole decision mechanism.

Interpreting the stacked result. The strongest hybrid number should be read carefully because the LLM components are evaluated on smaller subsets than the purely tabular baselines. Its value is methodological rather than a claim that the hybrid system dominates all alternatives: it shows that LLM-derived signals can correct some mistakes made by classical models, especially in the multiclass stage where class boundaries are noisier. A production system would need to re-run this comparison on the full inference budget and with stable API or local-model settings.

Why tabular baselines stay strong. The results are also a reminder that hashed enterprise tables remain a natural setting for tree ensembles. Target encoding and frequency features expose many of the regularities that matter for rare-event planning, while boosted trees can learn high-order interactions without relying on lexical meaning. This is the opposite of many text-centric LLM tasks: the most informative signal is often not in natural language, but in repeated categorical patterns, fleet counts, deadlines, and interactions between program metadata.

Operating point selection. For operational use, Stage 1 thresholds should be selected using precision–recall trade-offs rather than ROC-AUC alone. False positives waste preparation effort by triggering unnecessary retrofit planning, while false negatives create capacity risk when a vehicle appears without sufficient lead time. The high ROC-AUC values therefore need to be translated into plant-specific operating points, with thresholds chosen according to staffing, queue length, and the cost of missed preparation. This is also why PR-AUC and thresholded F1 are reported alongside ROC-AUC.

Deployment cost and temporal drift. The LLM components impose real deployment costs: embedding thousands of rows required minutes of API time, and prompted classification on a few hundred rows was slower still. When target encoding and boosted trees already perform strongly, those costs need a clear return. In this dataset, the strongest justification for LLM integration is complementary signal rather than raw leaderboard improvement. The temporal benchmark points in the same direction. Chronos-small and TIME-LLM provide usable forecasts without task-specific training, but the best lag-based models remain stronger on the short 76-point series. Prototype programs also change quickly: new derivatives, build phases, and retrofit packages appear over time. Any deployed system should therefore include drift monitoring, periodic retraining, and explicit checks for new hashed categories.

Using the predictions in practice. The three stages should be treated as decision-support signals rather than autonomous prescriptions. A high Stage 1 score can trigger early capacity reservation; a Stage 2 distribution can guide preparation for likely retrofit packages; and Stage 3 duration estimates can be combined with queue state to form a workload forecast. This staged use is important because uncertainty compounds across the pipeline. Keeping the outputs probabilistic and auditable makes it easier for planners to override questionable predictions and to collect feedback for retraining.

Conclusion. We presented a compact industrial planning study over enterprise tables. Across binary occurrence prediction, retrofit-type classification, duration regression, and monthly forecasting, the same message emerges: on privacy-constrained structured data, direct prompted classification is brittle, embedding features remain useful, and hybrid ML+LLM systems can improve selected tasks without displacing strong tabular baselines. Current LLM methods are thus most effective as complementary components for structured industrial prediction.

Limitations

The study uses internal industrial data and cannot provide a public benchmark with the same distribution. Although categorical values are hashed, the setup is still tied to one manufacturer, one planning process, and one retrofit department. LLM experiments are also subsampled because of cost and latency, so their results should be interpreted as deployment-oriented evidence rather than exhaustive model comparison. Finally, the monthly series has only 76 observations, which limits conclusions about time-series foundation models.

Ethical considerations

The work uses operational planning data with hashed categorical fields and excludes direct identifiers. Nevertheless, industrial prediction systems can influence staffing, scheduling, and prioritization decisions. Deployments should therefore include human oversight, monitoring for drift across new vehicle programs, and clear documentation of thresholds and failure modes. Prompted or API-based LLM components should be used only when privacy and contractual constraints allow row-level data transfer.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 2623–2631, New York, NY, USA. Association for Computing Machinery.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. 2024. Chronos: Learning the language of time series. *Preprint*, arXiv:2403.07815.

George Edward Pelham Box and Gwilym Jenkins. 1990. *Time Series Analysis, Forecasting and Control*. Holden-Day, Inc., USA.

Leo Breiman. 2001. Random forests. *Mach. Learn.*, 45(1):5–32.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. Smote: synthetic minority over-sampling technique. *J. Artif. Int. Res.*, 16(1):321–357.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA. Association for Computing Machinery.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. 2024. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org.

Tuan Dinh, Yuchen Zeng, Ruisu Zhang, Ziqian Lin, Michael Gira, Shashank Rajput, Jy-yong Sohn, Dimitris Papailiopoulos, and Kangwook Lee. 2022. Lift: language-interfaced fine-tuning for non-language machine learning tasks. In *Proceedings of the 36th International Conference on Neural Information Processing Systems*, NIPS '22, Red Hook, NY, USA. Curran Associates Inc.

Nick Erickson, Jonas Mueller, Alexander Shirkov, Hang Zhang, Pedro Larroy, Mu Li, and Alexander Smola. 2020. Autogluon-tabular: Robust and accurate automl for structured data. *Preprint*, arXiv:2003.06505.

Azul Garza, Cristian Challu, and Max Mergenthaler-Canseco. 2024. Timegpt-1. *Preprint*, arXiv:2310.03589.

Stefan Hegselmann, Alejandro Buendia, Hunter Lang, Monica Agrawal, Xiaoyi Jiang, and David Sontag. 2023. Tabllm: Few-shot classification of tabular data with large language models. *Preprint*, arXiv:2210.10723.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. 2023. Tabpfn: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations (ICLR)*.

Rob J. Hyndman and George Athanasopoulos. 2021. *Forecasting: Principles and Practice*, 3 edition. OTexts.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. 2024. Time-LLM: Time series forecasting by reprogramming large language models. In *International Conference on Learning Representations (ICLR)*.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: a highly efficient gradient boosting decision tree. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 3149–3157, Red Hook, NY, USA. Curran Associates Inc.

Daniele Micci-Barreca. 2001. A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *SIGKDD Explor. Newsl.*, 3(1):27–32.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. Catboost: unbiased boosting with categorical features. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 6639–6649, Red Hook, NY, USA. Curran Associates Inc.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, Marin Biloš, Sahil Garg, Anderson Schneider, Nicolas Chapados, Alexandre Drouin, Valentina Zantedeschi, Yuriy Nevmyvaka, and Irina Rish. 2024. Lag-llama: Towards foundation models for probabilistic time series forecasting. *Preprint*, arXiv:2310.08278.

Takaya Saito and Marc Rehmsmeier. 2015. The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets. *PLoS ONE*, 10(3):e0118432.

Sean J. Taylor and Benjamin Letham. 2018. Forecasting at scale. *The American Statistician*, 72(1):37–45.

A Additional dataset overview

Table 4: Dataset overview and derived learning tasks.

Dataset / task	Rows	Vehicles
Registration system (raw)	284,271	284,271
Retrofit system (cleaned)	48,716	11,830
Stage 1: binary (left join)	284,271	284,271
Stage 2: multiclass (inner join)	54,174	—
Stage 3: duration (inner join subset)	52,507	—

B Full classification tables

Tables 5 and 6 provide the complete binary and multiclass comparisons, including the stronger manual baselines omitted from the main text for space.

Table 5: Full binary occurrence results (Stage 1). F1 is reported at the optimal threshold.

Model / approach	ROC-AUC	PR-AUC	F1@opt	AUC CI
CatBoost (HP-tuned)	**0.997**	0.932	0.884	[0.997, 0.997]
Voting ensemble (top-3)	0.997	**0.937**	0.883	[0.997, 0.998]
Extra Trees (SMOTE)	0.997	0.935	0.880	[0.997, 0.998]
XGBoost (HP-tuned)	0.996	0.913	0.884	[0.996, 0.997]
LightGBM (HP-tuned)	0.996	0.910	0.883	[0.996, 0.997]
LLM-STACKED	0.996	0.897	0.900	[0.987, 1.000]
Random Forest (HP-tuned)	0.997	0.929	0.879	[0.996, 0.997]
LogReg (feat-selected)	0.990	0.810	0.777	[0.989, 0.991]
LLM-EMBED LogReg	0.982	0.667	0.684	[0.977, 0.986]
AutoGluon (Ens. L3)	0.997	—	0.881	—
LLM-PROMPTED (Sonnet 4)	0.500	0.045	0.086	[0.500, 0.500]

Table 6: Full multiclass retrofit-type results (Stage 2).

Model	Acc	$F1_w$	$F1_m$	AUC_{ovr}
AutoGluon (Ens. L3)	**0.702**	**0.654**	0.348	**0.934**
LLM-STACKED	0.680	0.626	0.281	0.888
Random Forest (SMOTE)	0.611	0.621	**0.390**	0.917
XGBoost (SMOTE)	0.601	0.614	0.380	0.914
LightGBM (SMOTE)	0.582	0.605	0.387	0.917
CatBoost (SMOTE)	0.589	0.600	0.356	0.906
LLM-EMBED LogReg	0.480	0.521	0.327	0.869
Logistic Regression (SMOTE)	0.293	0.332	0.148	0.768
LLM-PROMPTED (Sonnet 4)	0.100	0.018	0.012	0.500

C Monthly benchmark details

Table 7 reports the full monthly benchmark, including statistical baselines and the complete lag-feature comparison.

Table 7: Aggregated monthly time-series benchmark for retrofit duration. 76 observations (60 train, 16 test).

Family	Model	MAE	RMSE	R^2
Statistical baselines				
	SMA(3)	4.16	4.87	0.002
	SMA(6)	4.18	4.87	0.004
	SES	4.20	4.88	-0.001
	SBA	4.20	4.88	-0.002
	ARIMA(1,1,1)	4.19	4.91	-0.014
	Croston	4.22	4.93	-0.019
	ETS (additive)	4.24	5.82	-0.424
ML on lag features				
	LightGBM	**3.16**	**3.70**	-1.90
	AutoGluon	3.16	3.79	-1.88
	SGD	3.17	3.74	-1.74
	Ridge	3.22	3.80	-1.78
	XGBoost	3.33	4.00	-2.14
	MLP	3.39	3.90	-2.11
	Random Forest	3.44	3.99	-2.30
	LinearReg	3.48	4.11	-2.20
Foundation models				
	Chronos-small	4.03	5.04	-0.065
	Chronos-tiny	4.14	5.10	-0.091
	Chronos-base	4.25	5.32	-0.187
	TIME-LLM (frozen GPT-2)	4.62	5.56	-0.217
Ensemble				
	LightGBM + AutoGluon + SGD	3.16	—	-1.84