«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Центр непрерывного образования
Факультета компьютерных наук

ИТОГОВЫЙ ПРОЕКТ

Решение задачи регрессии для многокомпонентных временных рядов для детектирование режимов работы насосного оборудования.

Выполнил:

Сетдеков Кирилл Раильевич

Руководитель:

Абдуракипов Сергей Сергеевич

Москва 2021

Оглавление

Введение

В настоящее время на фоне бума развития свёрточных нейронных сетей в задачах обработки изображения и трансформеров для задач обработки естественного языка и перевода, ожидаемо что исследователи проводят эксперименты по использованию этих подходов для временных рядов. В частности, для задач прогнозирования многокомпонентных временных рядов сверточные сети, трансформеры[1] и подходы развивающие эти идеи дальше показали большой прогресс. Новые подходы вытеснили более традиционные для временных рядов модели с сезонностями или авторегрессионные модели из верхних строчек бенчмарков в последние 3 года. В этой работе я рассмотрю несколько из видов самых современных подходов (SOTA) и проанализирую их применимость к задаче прогнозирования параметров работы центробежных погружных насосов, используемых в нефтедобыче.

Обзор литературы

Для поиска наиболее современных подходов я использовал результаты сравнения моделей на сайте papers with code[1] по бенчмарку ETTh2 (720). Содержательно это часовые данные за 2 года, где измеряют качество прогноза на 30 дней, для данных по температуре трансформатора от 7 зависимых переменных. Этот бенчмарк наиболее сопоставим с теми данными, которые я использовал далее для анализа в части размерности и является наиболее сложным вариантом бенчмарка.

На основе сравнения методов по MAE на наборе данных ETTh2 (720), можно выделить 4 продвинутых подходов к прогнозированию временных рядом.

1https://paperswithcode.com/sota/time-series-forecasting-on-etth2-720

Перечисляю их в порядке снижения качества прогнозирования:

Ранг	Модель	MAE	MSE	Статья	Год
1	**SCINet** (Одномерная)	0.399	0.249	Time Series is a Special Sequence: Forecasting with Sample Convolution and Interaction	2021
2	**QuerySelector**	0.4130	0.2585	Long-term series forecasting with Query Selector – efficient model of sparse attention	2021
3	**Informer**	0.431	0.277	Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting	2020
4	**Transformer**	0.4340	0.2853	Long-term series forecasting with Query Selector – efficient model of sparse attention	2021
5	**SCINet** (Многомерная)	0.761	1.074	Time Series is a Special Sequence: Forecasting with Sample Convolution and Interaction	2021

1 и 5 место, отличаются тем, используется ли многомерная реализация или одномерная реализация модели. Подробнее остановимся на 4 подходах и трех статьях, где они описываются:

1. Time Series is a Special Sequence: Forecasting with Sample Convolution and Interaction[2] (использование свёрточных нейронных сетей и сравнение их с трансформерами) Данная статья проводит дальнейшее развитие актуальных архитектур для временных рядом (рекуррентные нейронные сети, модель транформеров, сверточные нейронные сети для временных рядов). Основная идея — модифицированный подход сверточной нейронной сети, где не только расширяется зона внимания

от слоя к слою, но и проводится последовательность сжатие-свертка-смешивание. Это позволяет одновременно смотреть на параметры временного ряда разного масштаба, что помогает лучше извлекать признаки из временного ряда и приводит к росту предсказательной способности в модели SCINet, предложенной авторами, в сравнении с прошлыми архитектурами.

На данный момент один из передовых по точности алгоритм, имеющий как многомерный так и одномерный вариант.

Есть официальный репозиторий[2] с кодом и реализацией на PyTorch.

2. Long-term series forecasting with Query Selector -- efficient model of sparse attention[3] (Использование трансформеров и query selector модели)

   В этой статье предлагается развития идеи трансформеров за счет построения детерминистического алгоритма построения разреженной матрицы внимания.

   Есть реализация кода от авторов модели на PyTorch[3].

3. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting[4] (использование информеров)

   Данная статья разбирает проблемы прогнозирования длительных временных рядов (LSTF) и отмечает проблемы использования архитектуры трансформеров для этих целей — сложность $O(N^2)$ по времени и большие требования по памяти и ограничения, присущие всем архитектурам энкодер-декодер. В качестве альтернативы, авторы предлагают более экономичную архитектуру модели для прогнозирования длительных временных рядов «Информеры», которые обладают тремя основными характеристиками 1) механизм внимания,

2 https://github.com/cure-lab/SCINet

3 https://github.com/moraieu/query-selector

который имеет сложность $O(N \cdot \log N)$по памяти и времени; 2) механизм внимания, который выделяет ключевые признаки, одновременно с делением пополам размера входов на каскадных слоях, что позволяет эффективно работать с чрезвычайно длинными временными рядами; 3) декодер генеративного типа, который строит прогноз за один шаг, а не итеративно, чем значительно ускоряет построение прогнозов.

В данной статье проводится сравнения с другими методами и показано превосходство предложенного подхода в разрезе прогнозирования длительных временных рядов над оригинальным вариантом использованием трансформеров.

Есть реализация кода от авторов модели на PyTorch[4].

Эти статьи про методы, которые не входят в state-of-the-art, но достойны упоминания:

1. Probabilistic Forecasting with Temporal Convolutional Neural Network[5]

   В статье описывается подход к созданию прогнозов для мультивариативных временных рядом на основе свёрточных нейронных сетей с использованием обучения признакам.

   Есть официальная реализация в коде от авторов работы на keras[5], что важнее — реализован внутри пакета Darts – современный и быстроразвивающийся пакет для работы с временными рядами[6]
2. N-BEATS: Neural basis expansion analysis for interpretable time series forecasting[6]

   Основной фокус на одномерных временных рядах — минус при

4 https://github.com/zhouhaoyi/Informer2020 — самый популярный по активности редакторов и наблюдателей репозиторий из этого обзора

5 https://github.com/ashishpatel26/tcn-keras-Examples

6 https://github.com/unit8co/darts

анализе мультивариантных временных рядов. Предложен и реализован подход на основе глубокой нейронной сети со связями вперед и назад между слоями и набором множества полносвязных слоев внутри архитектуры сети.

Есть реализация от компании, связанной с автором статьи[7]

Также реализован внутри пакета Darts

3. Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting[7].

   На этой работе в первую очередь базируется работа Liu 2021 года (модель SCINet)

   Основная идея — использование трансформеров и обучение поведению временного ряда на разных масштабах плюс выделение отдельной части модели под интерпретируемое представление внимания обученной сети, что позволяет и достигнуть лучших результатов в нескольких бенчмарках и отойти от реализации — черный ящик.

   Есть реализация на tensorflow[8], статья от исследователей в google research.

Методы

В рамках этой работы я планирую обучить 4 модели

- TCN;
- N-Beats;
- Transformer;
- Time Fusion Transformer.

7 https://github.com/ElementAI/N-BEATS

8 https://github.com/google-research/google-research/tree/master/tft

Плюс, как базовую — один из вариантов AutoML библиотек — LAMA, которая выбирает и обучает ансамбль бустингов на решающих деревьях с подбором гиперпараметров.

В части основных моделей, планируется использовать пакет Darts для обучения моделей TCN, N-Beats и трансформеров, и пакет PyTorch forecasting для модели Time Fusion Transformer.

В пакете Darts реализован загрузчик данных и обучающий цикл, основная сложность будет состоять в предобработке данных и формировании выборок в виде классов `darts.timeseries.TimeSeries`

Для модели Time Fusion Transformer необходимо будет настроить загрузчик данных из pandas.

Эксперименты

Описание данных

Для этой работы использованы данные, являющиеся сокращенным набором во времени и числу переменных набором данных, которые компания «Сургутнефтегаз» предоставляла в рамках открытого соревнования data science[8]. В работе использован набор данных по эксплуатации погружного оборудования фонда нефтяных скважин (17 скважин). Характеристики данных: 150 тысяч записей за два календарных месяца (июнь — июль 2019 года), выравненные по временной оси данные телеметрии с частотой дискретизации 5 минут. Для построения моделей использовали 15 независимых переменных, большая часть которых — телеметрия работы

насосов (нагрузка двигателя, коэффициент мощности, давление в коллекторе узла учета, наработка насоса, расход жидкости и газа и т.д.), а также идентификатор номера насоса и колонка времени изменений. Пример динамики целевой переменной, «средняя скорость изменения давления на приеме насоса в ЧАС, МПа/час.» показан на рисунке ниже.

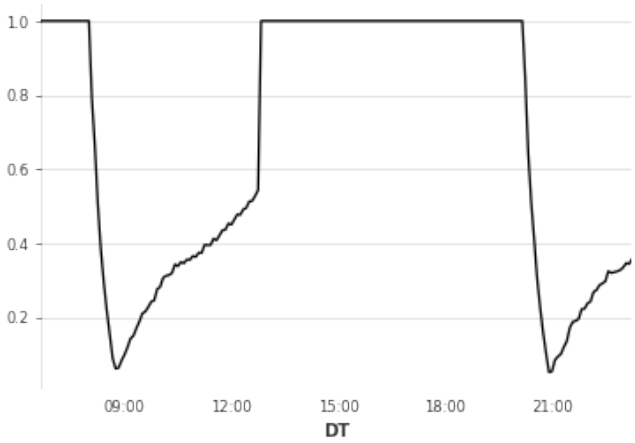


Рисунок 1: Нормированная к интервалу [0;1] динамика целевой переменной

Динамика признаков визуально выглядела хаотичной и не связанной с целевой переменной, за исключением признаков «T1138P6000096 Наработка двигателя с момента последнего включения, сек» и «T1138P4000064': "Загрузка двигателя, %», для остальных переменных направление влияние и степень их взаимосвязей первого порядка была заранее не очевидна. В качестве дополнительного анализа я использовал вывод feature importance для auto ml модели.

Ниже пример значений переменных:

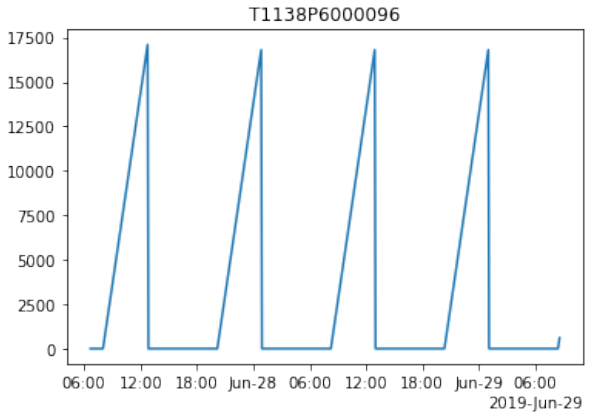


Рисунок 2: Время с момента включения насоса

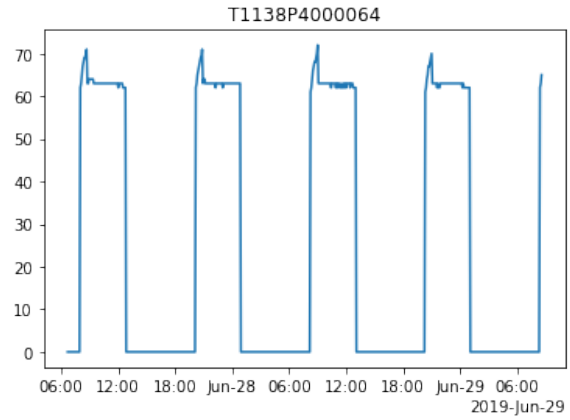


Рисунок3: Нагрузка насоса

P

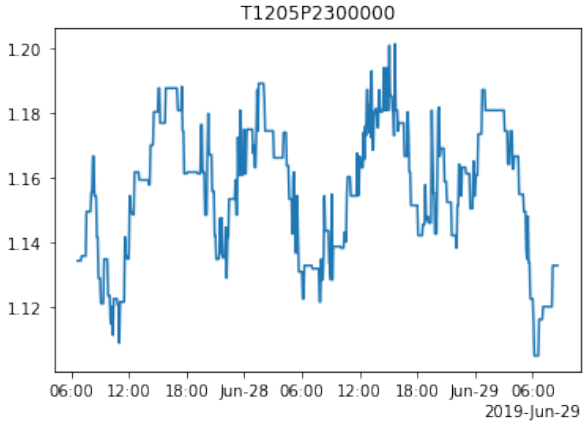


Рисунок 3: Средняя скорость изменения давления в коллекторе

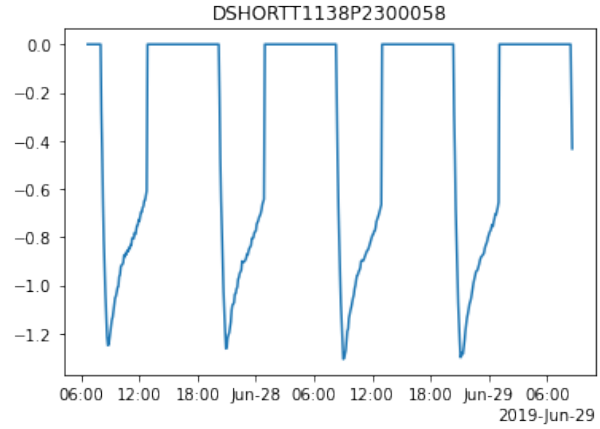


Рисунок 4: Целевая переменная

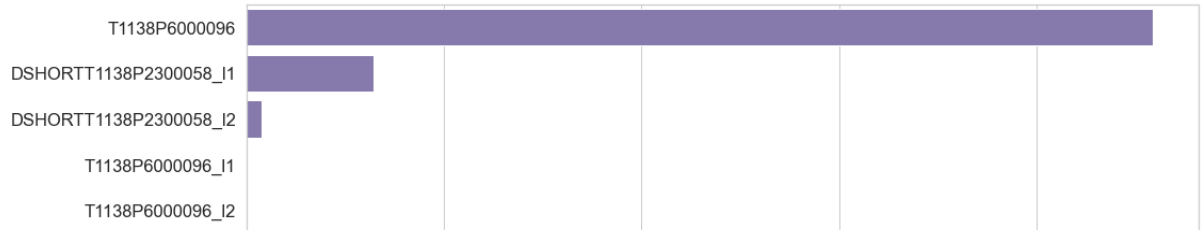


Рисунок 5: Нормированная значимость признаков для модели LAMA

На рисунке выше мы видим, что только 1 переменная (время включения насоса) и значение целевой переменной с задержкой 1 и 2 шага оказались значимые в AutoML модели. Можно предположить, что взаимосвязи между переменными нелинейные и отложенные во времени, что и требует применения современных методов машинного обучения, заточенных под

многомерные временные ряды.

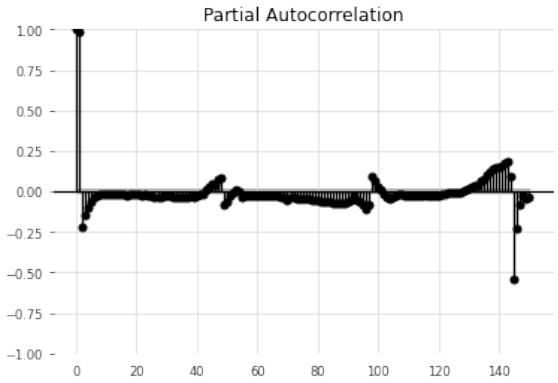


Рисунок 7: Частные автокорреляции для целевой переменной

Дополнительный анализ в части целевой переменной — анализ частных автокорреляций. Приведу правее график PACF для целевой переменной для случайного насоса. По нему видно, что до 144 лага значения являются значимыми. Уровень 144 временных интервала я буду использовать для моделей как базовую настройку размера входящего окна. Тесты показали, что более короткие значения приводят к снижению качества модели, а более высоки — увеличивают сложность моделей и увеличивают шанс переобучения.

Предобработка данных

Для AutoML модели из пакета LAMA значения целевой переменной и признаков не нормализовались, и идентификаторы насосов не кодировались отдельно — модель умеет работать с категориальными переменными. После того как модель показала плохой результат, я добавил значение целевой переменной и всех признаков с лагом 1 и 2.

Для всех остальных моделей использовался следующий поток обработки данных:

1. Нормализация значений всех переменных к интервалу [0,1];
2. Интерполяция пропусков в признаках вперед квадратичной функцией;
3. Кодирование id насосов с помощью LabelBinarizer в отдельные столбцы

со значениями 1 и 0;

4. Датафрейм из предыдущего шага был преобразован в список из датафреймов, где для каждого были наблюдения только по 1 насосу (это связано с ограничением пакета darts);
5. Для модели Time fusion transformer дополнительно:
    1. Создавалась колонка time_idx – индексы по времени, идущие от 0 с шагом 1 для каждого момента времени, который есть в обучающей выборке;
    2. Не проводилось разделение одного датафрейма на список из датафреймов, так как пакет PyTorch forecasting умеет работать с множественными наблюдениями на 1 момент времени;
    3. Был подготовлен dataset и dataloaded в формате PyTorch.

Построение моделей

На основе обучающих выборок, для целевой переменной «Средняя скорость изменения давления на приеме насоса в ЧАС, МПа/час» я построил 4 модели, описанных выше, которые я отбирал на основе баланса новизны и простоты применения:

- TCN;
- N-Beats;
- Transformer;
- Time Fusion Transformer.

Как базовую линию я использовал модель LAMA, которая является AutoML моделью, поддерживает автоматический подбор гиперпараметров и обучает ансамбль из бустингов в рамках фиксированного времени на обучение.

Общий подход к построению и тестированию моделей:

- В моделях использовались все независимые переменные;

- Прогноз строился на 1 шаг вперед;
- Для сравнения модели использовались 4 метрики:
  - SMAPE (основная метрика для ранжирования моделей) — симметричная абсолютная ошибка в процентах. Выбрана основной на основе недостатка MAPE – при равенстве истинного значения 0, MAPE выдает неопределенные значения, так как происходит деление на 0. Специфика данных — в целевой переменной присутствуют длительные периоды нулевых значений.
  - MAE – средняя абсолютная ошибка;
  - R2 – коэффициент детерминации.
- Гиперпараметры были установлены значениями по умолчанию, или подобраны вручную для максимизации SMAPE на валидационных выборках;
- Обучение модели шло на 70% первых наблюдений из обучающей выборки, на 30% подбирались гиперпараметры, потом модель обучалась на всей обучающей выборке;
- Валидация и финальные метрики для результатов модели проводились на двух отложенных выборках для валидации «valid1.csv» и «valid2.csv». Прогноз моделей на двух выборках и целевая переменная конкатенировались по двум валидацоинным выборкам и на этих данных рассчитаны метрики.

С подробным кодом с реализацией всех методов можно ознакомится на GitHub: https://github.com/ksetdekov/ts_transformers

Результаты построения моделей

Финальные результаты работы моделей на выборке для валидации приведены в таблице ниже:

	SMAPE	MAE	R2
TCN	0,04443	0,01243	0,97863
Transformer	0,07002	0,02558	0,95225
N-Beats	0,17577	0,08316	0,81066
LAMA AutoML	1,52328	0,05426	0,95214
Time Fusion Transformer	1,55919	0,03595	0,89242

Для трех из использованных подходов удалось получить SMAPE лучше, чем для AutoML подхода, при этом лучший результат получен для модели TCN.

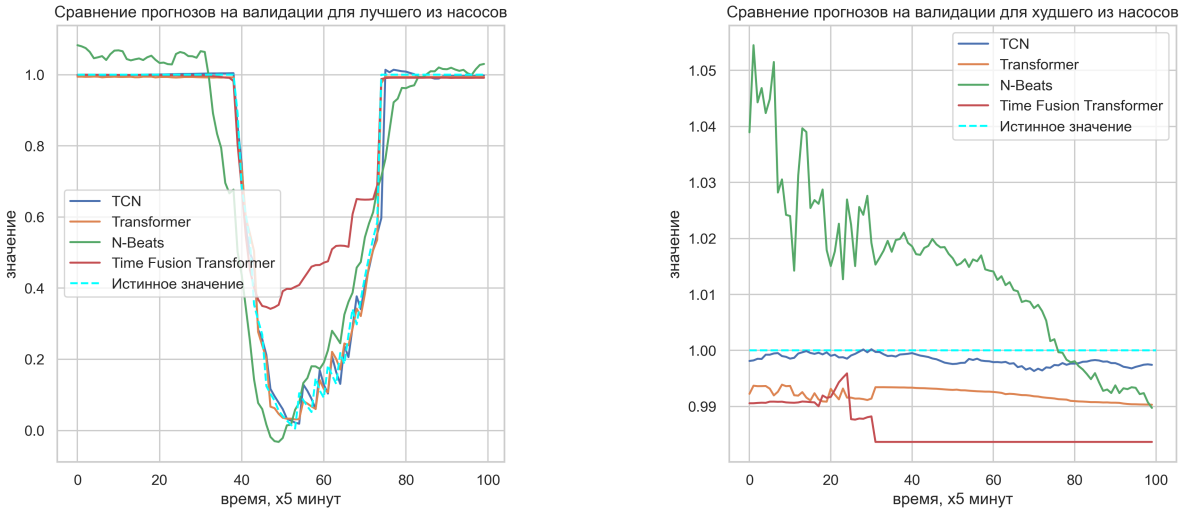


Рисунок 6: Пример прогноза моделей в сравнении с истинным значением

На графиках выше, мы видим визуальное сравнение качество прогноза моделей для лучшего и для худшего насоса в валидационной выборке. Пример лучшего прогноза для модели TCN приведен ниже. Мы можем увидеть, что предсказание крайне точно, с SMAPE < 0,014 и MAE < 0,005 для насоса №8

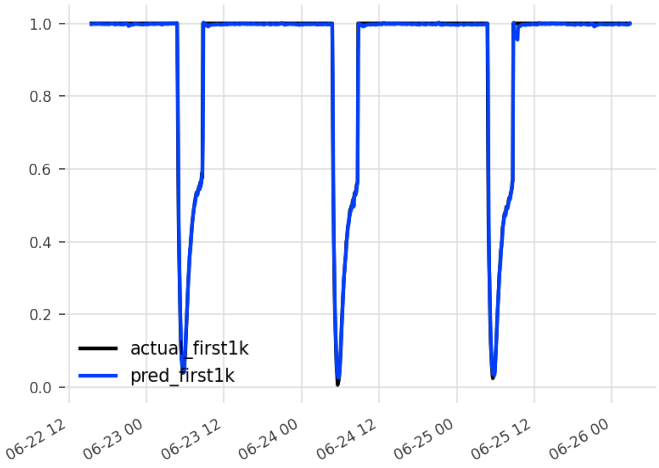
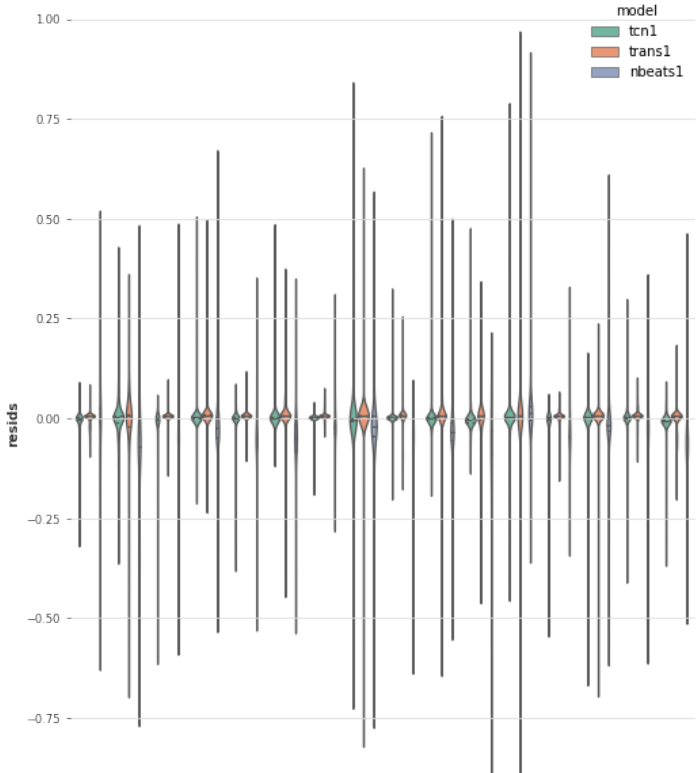


Рисунок 7: Факт и прогноз на 1000 наблюдений для модели TCN на валидационной выборке

Сравнение графиков остатков приведено в приложении на странице 20. К анализу метрик качества моделей, интерпретация этих графиков может добавить только два момента:

1. Все методы нелинейные, остатки распределены крайне далеко от нормального распределения.
2. Мы видим, что для всех методов, дисперсия остатков выше для максимальных значений, что применимо к данным означает, что решить задачу определения начала аномалии для модели тяжелее, чем предсказать целевую переменную, когда отклонение уже началось.



Анализ остатков в разрезе по насосам, приведенный на графике слева только подтверждает наблюдения, которые сделаны без разреза по насосам — для лучшей модели, TCN разброс остатков в среднем

eats. При этом для каждого насоса можно отметить, что распределения имеют тяжелые хвосты и высокую концентрацию около 0. Систематических отклонений от прогноза для какого-либо насоса не выявлено.

Заключение

1. В рамках этой работы удалось показать, что SOTA модели для многокомпонентных временных рядов работают лучше AutoML подходов (3 из 4 моделей показали лучшее качество на валидации).
2. На основе анализа эмбеддинга catch22 для временных рядов удалось найти связь между характеристиками временных рядов, где модели работали хуже (в приложении 2, ниже). Интерпретируемый вывод из этого — можно оценивать отклонение распределения целевой переменной в прогнозах и таким образом детектировать заранее снижение качества прогнозов. Также, анализ 3 компонент: DN_HistogramMode_5, DN_HistogramMode_10 и CO_HistogramAMI_even_2_5 отвечающих за форму распределения указывает на то, что хуже всего методы Трансформов и N-Beats сработали для групп насосов, где в обучающей выборке было меньше отклонений от константного значения. Потенциально решением может быть проведение балансировки обучающей выборки или накопления большего объема данных.
3. Модель time fusion transformer на доступных данных имела тенденцию к переобучению, что можно связать с большим числом весов даже в базовой модели (160 тысяч весов и 140 тысяч наблюдений).
4. Для моделей трансформеров, временные ряды, по которым лучше всего работал прогноз были похожи на данные из области финансов, а ряды с

худшим качеством на валидационной выборке — на временные ряды из области микроэкономики, согласно сравнению результатов из базы данных CompEngine[9].

Приложения

Приложение 1. Гиперпараметры для моделей

LAMA

Финальная модель обучалась методом TabularUtilizedAutoML, который подбирает гиперпараметры и строит ансамбли из моделей отталкиваясь от ограничения по времени. Я использовал:

- timeout (ограничение по времени) – 1000 секунд
- cpu_limit (число ядер) – 10
- cv (кросс-валидация) — 5

TCN

Финальная модель TCNModel из пакета Darts была обучена со следующими параметрами:

- n_epochs (число эпох) — 100;
- input_chunk_length (длина входящего окна) — 144;
- output_chunk_length (длина прогноза) — 1;
- dropout (вероятность dropout на сверточных слоях) — 0.01;
- dilation_base (основание для степенной функции, которая отвечает за расширение рецептивного поля на каждом слое) 2;
- weight_norm (использовать нормализацию весов) True;

9 https://www.comp-engine.org/

- kernel_size (размер ядра сверки) – 5;
- num_filters (число каналов на каждом слое сверки) — 3.

N-Beats

Финальная модель NBEATSModel из пакета Darts была обучена со следующими параметрами:

- n_epochs (число эпох) — 100;
- input_chunk_length (длина входящего окна) — 144;
- output_chunk_length (длина прогноза) — 1;
- generic_architecture=True,
- num_stacks (число слоев трендов и сезонностей) — 10,
- num_blocks (число блоков в одном слое) — 1,
- num_layers (число полносвязных слоев в каждом блоке каждого стэка) — 4,
- layer_widths (число нейронов в полносвязных слоях) — 512,
- nr_epochs_val_period (сколько эпох ждать до начала теста) —1,
- batch_size (размер батча) —800.

Transformer

Финальная модель TransformerModel из пакета Darts была обучена со следующими параметрами:

- n_epochs (число эпох) — 100;
- input_chunk_length (длина входящего окна) — 144;
- output_chunk_length (длина прогноза) — 1;
- dropout (вероятность dropout на сверточных слоях) — 0.1;
- batch_size (размер батча) — 32;
- d_model (ожидаемое число признаков) — 32;

- nhead (число голов на входе механизма внимания) — 16;
- num_encoder_layers (число слоев энкодера) — 3;
- num_decoder_layers (число слоев декодера) —3;
- dim_feedforward — 512;
- функция активации — "relu";
- остальные параметры — стандартные по-умолчанию

Time Fusion Transformer

Финальная модель TemporalFusionTransformer из пакета pytorch_forecasting была обучена со следующими параметрами:

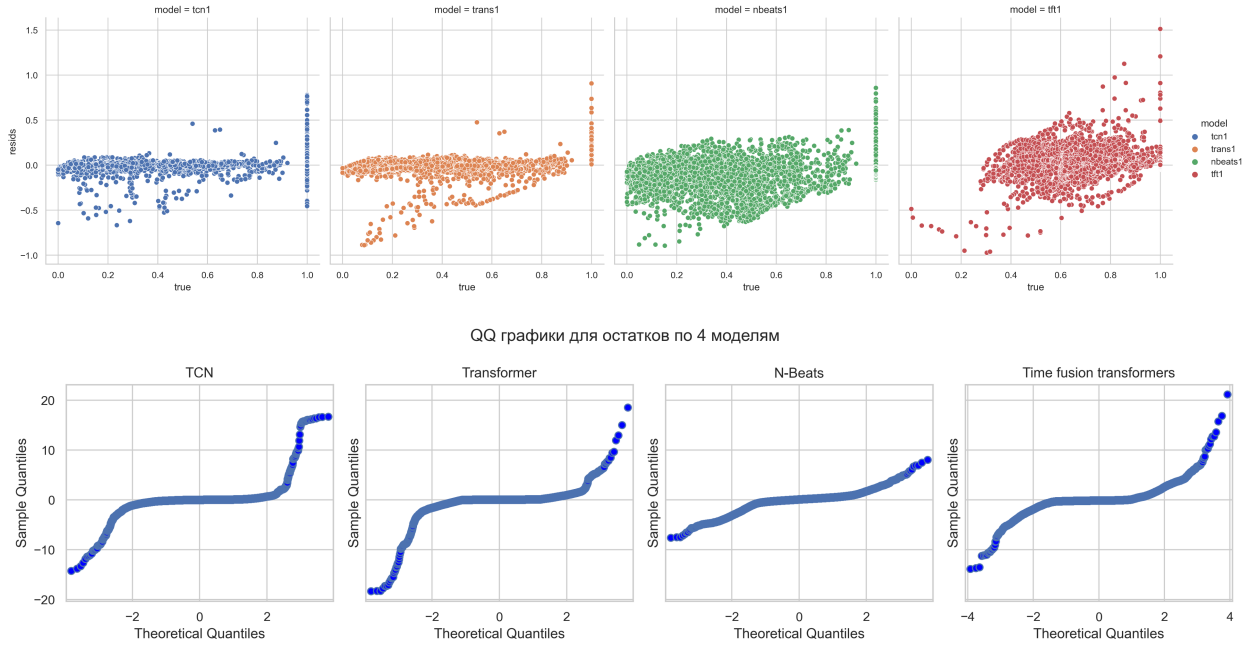
- learning_rate (коэффициент скорости обучения) — 0,1789, подобран на обучающей выборке;
- размер скрытого слоя — 32;
- dropout (вероятность dropout на сверточных слоях) — 0.1;
- остальные параметры — стандартные по-умолчанию.

Приложение 2.

p-value для теста на равенство средних значений соответствующего эмбеддинга из Catch22 в группах насосов, где метод работал выше и ниже медианы по SMAPE

Компонент эмбеддинга Catch22	Transformer	TCN	N-Beats	Time fusion transformer
DN_HistogramMode_5	0.015071	0.199686	0.010161	0.111029
DN_HistogramMode_10	0.012908	0.194820	0.008116	0.111845
CO_f1ecac	0.120809	0.685683	0.088088	0.047290
CO_HistogramAMI_even_2_5	0.034984	0.327320	0.016642	0.123993
SB_TransitionMatrix_3ac_sumdiagcov	0.009079	0.146661	0.009079	0.615027
CO_Embed2_Dist_tau_d_expfit_meandiff	0.107719	0.627726	0.077979	0.025494

Приложение 3. график остатков модели в зависимости от истинного значения и Q-Q графики распределения остатков



Список литературы

1: Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, Attention is All You Need, 2017
2: Minhao Liu, Ailing Zeng, Zhijian Xu, Qiuxia Lai, Qiang Xu, Time Series is a Special Sequence: Forecasting with Sample Convolution and Interaction, 2021
3: Jacek Klimek, Jakub Klimek, Witold Kraskiewicz, Mateusz Topolewski , Long-term series forecasting with Query Selector -- efficient model of sparse attention, 2021
4: Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, Wancai Zhang , Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting, 2021
5: Yitian Chen, Yanfei Kang, Yixiong Chen, Zizhuo Wang, Probabilistic Forecasting with Temporal Convolutional Neural Network, 2020
6: Boris N. Oreshkin, Dmitri Carpov, Nicolas Chapados, Yoshua Bengio, N-BEATS: Neural basis expansion analysis for interpretable time series forecasting, 2020
7: Bryan Lim, Sercan O. Arik, Nicolas Loeff, Tomas Pfister, Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting, 2020
8: S Abdurakipov, On increasing the efficiency of electric submersible pumps by using big data processing technologies,