DOI: 10.18137/RNU.V9187.22.04.P.62 УДК 004.85

В.А. Канарский

ПРОГНОЗИРОВАНИЕ ОТКАЗОВ НАСОСНОЙ СТАНЦИИ С ПОМОЩЬЮ МАШИННОГО ОБУЧЕНИЯ БЕЗ УЧИТЕЛЯ

Аннотация. Машинное обучение лежит в основе многих инновационных технологий искусственного интеллекта. Внедрение программ на основе машинного обучения в производство позволяет предсказывать поломки промышленного оборудования, предотвращая огромные расходы на оперативное техническое обслуживание. На основе временных данных датчиков насосной станции исследована эффективность различных алгоритмов машинного обучения без учителя на базе библиотеки Scikit-Learn Python.

Ключевые слова: машинное обучение без учителя, Scikit-learn, прогностическое обслуживание, датчики, временные ряды, гауссовы смеси, локальный фактор выброса, изолирующий лес.

V.A. Kanarsky

PREDICTING PUMPING STATION FAILURES USING UNSUPERVISED MACHINE LEARNING

Abstract. Machine learning is at the heart of many innovative artificial intelligence technologies. Implementation of machine learning-based programs in production allows predicting breakdowns of industrial equipment, thus preventing huge operational maintenance costs. Based on temporal data from pumping station sensors, the effectiveness of various teacherless machine learning algorithms based on the Scikit-Learn Python library is investigated.

Keywords: unsupervised machine learning, Scikit-learn, predictive maintenance, sensors, time series, Gaussian mixtures, Local outlier factor, Isolation Forest.

Введение

С приходом науки о данных (от англ. Data Science) появилось множество путей решения проблем в самых различных отраслях – экономике, науке, производстве и др. Начиная от сортировки спама на почте и заканчивая роботами, управляемыми искусственными нейронными сетями, Data Science все больше охватывает современный мир. Однако если в каких-то областях, например, финансовой индустрии с ее скорингом клиентов, динамическим ценообразованием и подобными задачами искусственный интеллект уже прочно закрепился в качестве основного инструмента, то про его внедрение в промышленность так сказать пока нельзя. Тем не менее оптимизировать обработку и анализ потока данных технологических процессов вполне возможно. Конечно, в таком объеме эти данные не будут нужны человеку, но могут оказаться полезны для моделей машинного обучения, потому что именно они решают задачу *прогностического обслуживания* (от англ. Predicted Maintenance).

Производственная сфера считается тяжелой отраслью, в которой используются различные виды тяжелого оборудования, такие как двигатели, насосы, трубы, поезда и др., которые всегда рассматриваются как наиболее важные ресурсы для работы компании. Поэтому целостность и надежность этого оборудования являются основными направлениями в их программах управления активами и определяются различными видами технического обслуживания.

Канарский Вадим Андреевич
аспирант, преподаватель кафедры автоматики телемеханики и связи. Дальневосточный государственный университет путей сообщения; город Хабаровск. Сфера научных интересов: автоматика и телемеханика на железнодорожном транспорте; машинное обучение и искусственный интеллект; прикладная математика. Автор 2 опубликованных научных работ. Электронный адрес: jizzierose@yahoo.com

Практически всегда проще выполнять плановый ремонт, однако он характеризуется огромными эксплуатационными и трудовыми затратами. Внедрение диспетчерских систем контроля на основе номинальных параметров и их допустимых границах позволяет производить ремонт по состоянию, исключая аварийные отказы. Но именно прогностическое обслуживание дает возможность определять аномальное поведение устройств, даже если их характеристики не выходят за пределы. Поэтому наличие возможности заранее обнаруживать аномалии и уметь снижать риски является очень ценной способностью, позволяющей предотвращать незапланированные простои и ненужное техническое обслуживание, которые выражаются в серьезных суммах в денежном эквиваленте.

Для решения данной задачи применяются методики из таких областей искусственного интеллекта, как *машинное обучение* (Machine Learning) и *глубокое обучение* (Deep Leraning). Машинное обучение подразделяется на обучение с учителем (Supervised Learning), без учителя (Unsupervised Learning), частичным привлечением учителя (Semi-Supervised Learning) и подкреплением (Reinforcement Learning).

В данной статье описываются этапы подготовки, выбор и обучение подходящих моделей на примере набора данных (Dataset) датчиков насосной станции [8], а также план работы с такими данными и сравнительная характеристика нескольких моделей.

Разведочный анализ данных. Очистка

В качестве инструмента для анализа данных использован язык программирования Python, а также специализированные библиотеки Numpy, Pandas, Matplotlib, Scikit-Learn.

Загруженный набор данных представляет собой файл с расширением CSV, где информация записана в текстовом формате. Структура кадра данных (Data Frame) представлена на Рисунке 1. В нем 55 столбцов (признаков) и 220320 строк (экземпляров). Согласно первому столбцу timestamp имеются поминутно зафиксированные числовые данные 52 датчиков и состояния, в котором пребывала насосная станция – нормальном (Normal), неисправном (Broken) и восстановленном (Recovering). Таким образом, имея информацию от timestamp, можно говорить об анализе временных рядов.

На Рисунке 2 представлен срез данных нескольких экземпляров в начале, середине и конце. Можно заметить, что сенсор № 50 имеет особые значения Nan – пустые ячейки, которые обязательно требуют обработки, так как большинство моделей машинного обучения не работают с пустыми значениями.

Начальным этапом при работе с подобными данными является их очистка, а также удаление дубликатов, низкоинформативных столбцов, заполнение пропусков [2]. К примеру, по датчику № 15 нет никаких данных, поэтому он был удален из исследуемого датафрейма.

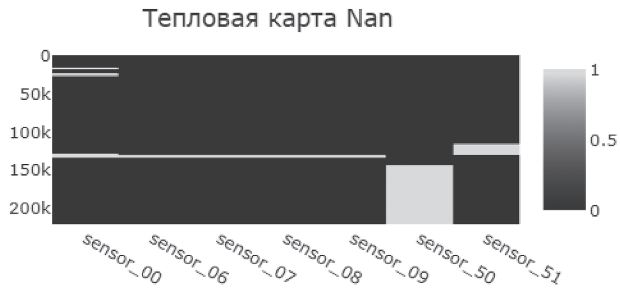
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 220320 entries, 0 to 220319
Data columns (total 55 columns):
 #   Column          Non-Null Count   Dtype
---  ------          --------------   -----
 0   Unnamed: 0      220320 non-null  int64
 1   timestamp       220320 non-null  object
 2   sensor_00       210112 non-null  float64
 3   sensor_01       219951 non-null  float64
 ...    ...             ...      ...      ...
 17  sensor_15       0 non-null       float64
 ...    ...             ...      ...      ...
 52  sensor_50       143303 non-null  float64
 53  sensor_51       204937 non-null  float64
 54  machine_status  220320 non-null  object
```

Рисунок 1. Структура датафрейма

	timestamp	sensor_00	sensor_01	sensor_02	...	sensor_50	sensor_51	machine_status
0	2018-04-01 00:00:00	2.465394	47.09201	53.211800	...	243.0556	201.3889	NORMAL
1	2018-04-01 00:01:00	2.465394	47.09201	53.211800	...	243.0556	201.3889	NORMAL
2	2018-04-01 00:02:00	2.444734	47.35243	53.211800	...	241.3194	203.7037	NORMAL
...	...	...	...	...	...	...	...	...
77790	2018-05-25 00:30:00	2.321759	47.482640	51.475693		220.1968	267.3611	BROKEN
...	...	...	...	...	...	...	...	...
220315	2018-08-31 23:55:00	2.407350	47.69965	50.520830	...	NaN	231.1921	NORMAL
220316	2018-08-31 23:56:00	2.400463	47.69965	50.564240	...	NaN	231.1921	NORMAL

Рисунок 2. Датафрейм

Судя по Рисунку 3, удалению подлежит sensor_50, так как он имеет большое количество NaN. Оставшиеся датчики образуют небольшую полосу отсутствующих значений в одно и то же время. Можно предположить, что произошла какая-то аварийная ситуация, и большинство сенсоров отключились. Тем не менее они не были удалены, их пропуски были заполнены средним значением по столбцам для корректной работы моделей. Помимо этого (см. Рисунок 4) sensor_00 имел наименьшую дисперсию, соответственно, вряд ли принес бы пользу при обучении моделей, поэтому он также был удален из датафрейма.



	Процент пропусков
sensor_50	34.956881
sensor_51	6.982117
sensor_00	4.633261
sensor_07	2.474129
sensor_08	2.317992
sensor_06	2.177741
sensor_09	2.085603

Рисунок 3. Тепловая карта датчиков, имеющих процент пропусков больше 2

	Ср.квадр. отклонение
sensor_00	0.412227
sensor_18	0.765883
sensor_08	2.037390
...	...
sensor_31	283.544760
sensor_36	289.385511

Рисунок 4. Среднеквадратичное отклонение зафиксированных значений датчиков

Исследование закономерностей

Графическое отображение временных рядов позволило выявить тенденции и любые странности в поведении значений. В частности, имело смысл взглянуть на показания датчиков, отображаемых во времени, и состояния машин BROKEN (показано крестиками) и RECOVERING (показано точками). Таким образом, на Рисунке 5 можно четко проследить, когда насос ломается, и как это отражается на показаниях на примере датчика № 5.

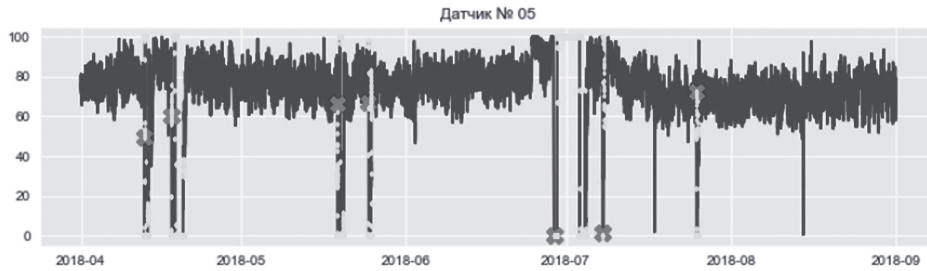


Рисунок 5. Временная характеристика датчика № 5

Из приведенного выше графика крестики и точки, отражающие, соответственно, неисправное и восстанавливаемое состояния насоса, полностью совпали с наблюдаемыми отклонениями показаний датчика. При этом есть непомеченные пиковые значения сенсора, которые могут свидетельствовать о предотказном состоянии насосной станции.

Стационарность и автокорреляция

При анализе временных рядов важно, чтобы данные были *стационарными* и не имели *автокорреляции*. Стационарность подразумевает, что среднее значение и стандартное отклонение данных не изменяются со временем, иначе ряд считается нестационарным.

Данные на Рисунке 6 выглядят стационарными; среднее и стандартное отклонения слабо менялисьются со временем за исключением времени простоя насоса, что вполне ожидаемо. Так обстоят дела с большинством датчиков в этом наборе данных. Однако такое может быть не всегда, и в таких ситуациях необходимо применять различные методы преобразования, чтобы сделать данные стационарными перед обучением моделей.

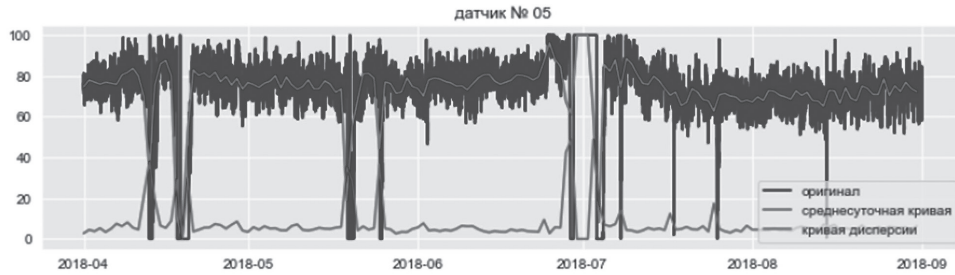


Рисунок 6. Временной ряд с кривыми среднего и среднеквадратичного отклонения в сутки

Изучение временного ряда означает исследование трех его компонент: тренда, сезонности и случайности. Последний компонент – это и есть разница (остаток или ошибка) между прогнозируемыми и реально наблюдаемыми значениями. Последовательная корреляция (или автокорреляция, далее – АКФ) показывает, насколько сильно коррелируют эти ошибки. Это крайне важно для достоверности предсказаний модели, поскольку АКФ неразрывно связана со стационарностью, так как, по определению, стационарные временные ряды последовательно некоррелированы (неавтокоррелированы). Игнорирование этого факта будет означать, что предсказания нашей модели будут неверными [1].

Исследование автокорреляции данного ряда представлено на коррелограмме (см. Рисунок 7) каждые 5760 мин. – 4 дня (выбрано из удобства отображения). На лаге 0 коррелограмма всегда имеет значение 1 (корреляция наблюдений с самими собой).

На Рисунке 7 отчетливо видно, что почти все лаги находятся внутри закрашенной области – доверительном интервале (5 %), что означает, что эти значения АКФ статистически незначимы. Однозначно убедиться в этом утверждении позже позволит тест Дики – Фуллера [3].

Предобработка и понижение размерности

Обучение моделей с использованием 50 датчиков потребует значительных вычислительных затрат и не является эффективным. К тому же одни характеристики более постоянны, чем другие, имеющие большую дисперсию (см. Рисунок 4). Это приводит к тому, что экземпляры данных будут лежать в пределах гораздо более низкоразмерного подпространства 50-мерного пространства. *Метод главных компонент* PCA является самым популярным алгоритмом уменьшения размерности. Сначала он определяет гиперплоскость, которая лежит ближе всего к данным, а затем проецирует данные на нее, создавая новые характеристики, которые будут использоваться для моделирования, стараясь при этом сохранить большую часть дисперсии [3].

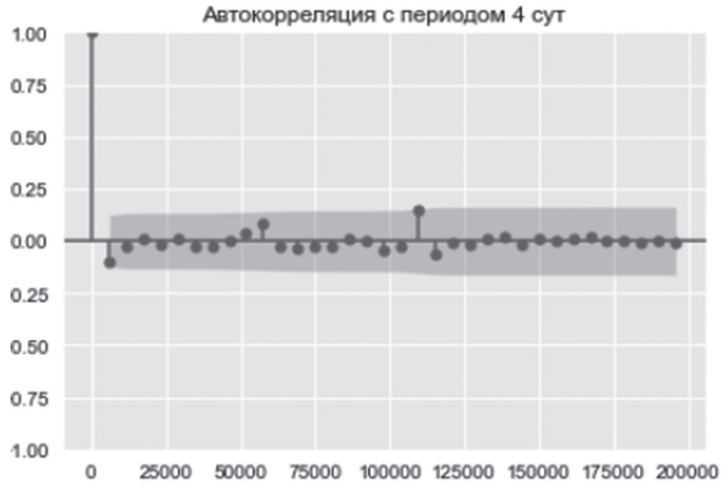


Рисунок 7. Коррелограмма датчика № 5

Для правильного применения PCA данные должны быть масштабированы и стандартизованы. Это связано с тем, что PCA и большинство алгоритмов обучения – это алгоритмы, основанные на оценке расстояния. Если обратить внимание на первые несколько строк данных (см. Рисунок 2), величины значений каждого признака сильно разнятся. Некоторые из них очень малы, в то время как другие имеют большие значения. Стандартизация позволит привести значения к единому диапазону – от –1 до 1 по формуле

$$z = \frac{x_0 - \overline{x}}{\sqrt{\sigma^2}}, \quad (1)$$

где x_0 – значения (экземпляры) одного столбца (датчика); $\overline{x}$ – среднее значение по столбцу; σ – среднеквадратичное отклонение по столбцу.

После стандартизации и применения метода главных компонент можно увидеть, сколько информации содержит каждая компонента.

Согласно Рисунку 8 первые две главные компоненты являются наиболее важными в соответствии с теми, которые были извлечены с помощью PCA на приведенном выше графике значимости [4]. Их совокупный процент объясненной дисперсии составляет 35,6 + 18,6 = 54,2 %, то есть больше половины информации несут именно они. Тем самым многомерный набор данных можно отразить в низкоразмерный 2D.

Для сохранения большей части дисперсии обучения имело бы смысл взять первые семь главных компонент, однако визуализировать это будет гораздо тяжелее, тем более что после применения PCA страдает интерпретируемость данных.

Дополненный тест Дики – Фуллера (тест ADF)

Чтобы окончательно убедиться в стационарности процессов и отсутствии автокорреляции, был проведени тест ADF с результатами проверки нулевой и альтернативной гипотез. На основе специального *p*-критерия (*p*-уровень значимости) можно сделать вывод о стационарности временного ряда. Тест выполнен при 1,5- и 10%-м уровнях значимости для двух главных компонент [4].

Рисунок 8. Распределение дисперсии между компонентами PCA

```
Критерий ADF: -4.577571780810939
P-значение: 0.00014203951221862258
Критические значения:
1% -3.430379692585317
Нулевая гипотеза отвергнута – Временной ряд стационарен
5% -2.8615531237364724
Нулевая гипотеза отвергнута – Временной ряд стационарен
10% -2.5667769852919347
Нулевая гипотеза отвергнута – Временной ряд стационарен
```

Рисунок 9. Результаты теста

Для 1-й главной компоненты получен *p*-уровень значимости 0,00014 (0,014 %), что намного меньше 1 %. Таким образом, нулевая гипотеза была отвергнута – данные стационарны, и с течением времени *t* менее коррелированы (то есть с ростом *t* временной ряд «забывает» свои прошлые состояния) [1]. Результаты теста аналогичны в случае со второй компонентой.

Моделирование

Детектирование аномалий, как правило, является задачей обучения без учителя, так как изначально предполагается, что аномалии редки и отличаются от нормальных точек [7], поэтому, даже имея маркированные данные Broken нет возможности использовать алгоритмы обучения с учителем, потому что этих данных слишком мало (7 из 220320).

Определившись с видом алгоритма машинного обучения и получив более простое отображение информации всех датчиков (см. Рисунок 10), можно предположить определенные дальнейшие алгоритмы.

Гауссовы смеси (Gaussian Mixture)

Форма кластера точек напоминает два ортогональных эллипсоидных облака, каждое из которых гипотетически имеет свое нормальное распределение. Плотность облаков снижается по мере удаления от своих главных полуосей. Аномальные точки в этом случае те, что лежат в областях с низкой плотностью.

Вычислить параметры этих распределений позволила модель *гауссовых смесей*, в которой заранее предполагался тот факт, что имелись две нормально распределенные случай-

ные величины. Атрибутом модели coverence_type='diag' из кластера точек выделяются эллипсоиды [10].

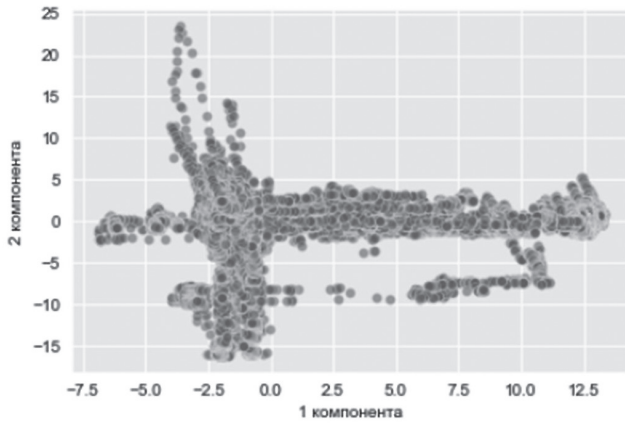


Рисунок 10. Диаграмма рассеяния двух главных компонент

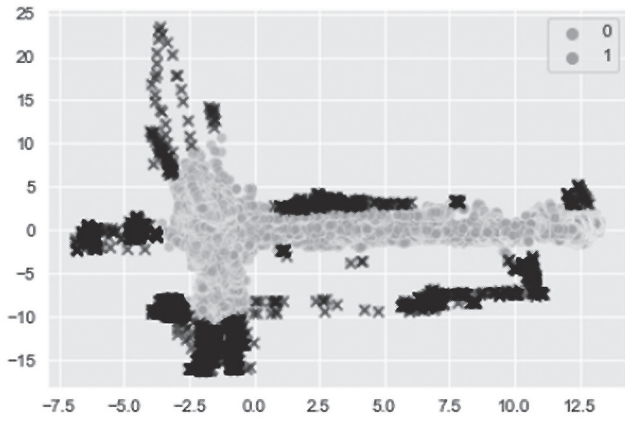


Рисунок 11. Диаграмма рассеяния двух гауссовых распределений и их аномальные точки

На Рисунке 11 метки 0 и 1 принадлежат двум нормально распределенным случайным величинам. Алгоритм выявил 7051 аномальных точек (показаны черным). Соотнося их с показаниями датчика, получается картина как на Рисунке 12.

В результате обработки этих данных рассчитано крайнее время предупреждения перед аварией. Минимальным значением порога, при котором каждая из поломок была спрогнозирована, стал порог в 3,2 %.

Локальный фактор выброса (Local Outlier Factor (LOF))

Данный алгоритм специализирован для обнаружения выбросов данных и как гауссовы смеси опирается на плотность точек, однако реализуется с противоположной позиции (см. Рисунок 13).

LOF определяет выброс на основе локального соседства с учетом относительной плотности окружения [6] и идентифицирует аномальные точки, не учитывая, в отличие от га-

уссовых смесей, глобальное распределение. Использование манхэттенской нормы вместо евклидовой дает результат 1019 обнаруженных выбросов против 992 (см. Рисунок 14).


```
Последнее предупреждение до поломки №1 2018-04-12 было за 3 days 15:28:00
Последнее предупреждение до поломки №2 2018-04-18 было за 4 days 07:22:00
Последнее предупреждение до поломки №3 2018-05-19 было за 8 days 00:04:00
Последнее предупреждение до поломки №4 2018-05-25 было за 0 days 00:01:00
Последнее предупреждение до поломки №5 2018-06-28 было за 0 days 00:01:00
Последнее предупреждение до поломки №6 2018-07-08 было за 3 days 08:54:00
Последнее предупреждение до поломки №7 2018-07-25 было за 1 days 13:18:00
```

Рисунок 12. Показания датчика № 5 с выявленными аномалиями с помощью гауссовых смесей и крайним временем предупреждения

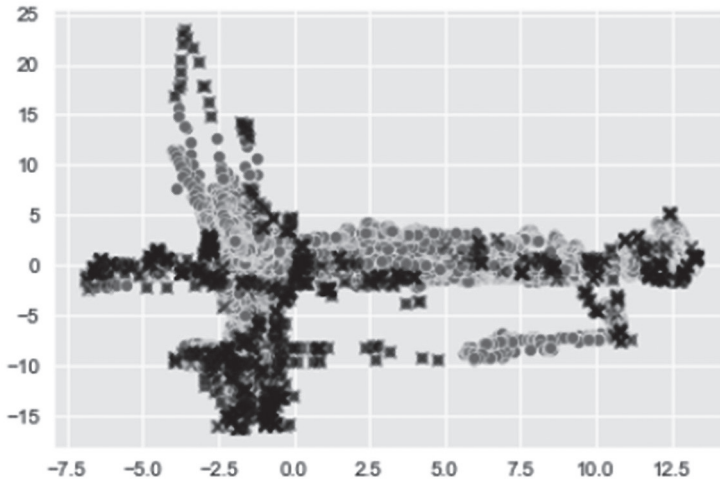


Рисунок 13. Детектирование аномалий с помощью LOF

Изолирующий лес (от англ. Isolation Forest)

Данный алгоритм, основанный на древовидной структуре, является одним из основных для решения задач по детектированию аномалий (см. Рисунок 15).

Этот алгоритм не моделирует нормальные точки, а определяет изолированность наиболее выбивающихся, аномальных. Алгоритм эффективен для обнаружения выбросов, особенно в массивах данных с высокой размерностью.


```
Последнее предупреждение до поломки №1 2018-04-12 было за 2 days 04:17:00
Последнее предупреждение до поломки №2 2018-04-18 было за 3 days 21:51:00
Последнее предупреждение до поломки №3 2018-05-19 было за 2 days 15:30:00
Последнее предупреждение до поломки №4 2018-05-25 было за 0 days 00:01:00
Последнее предупреждение до поломки №5 2018-06-28 было за 0 days 00:59:00
Последнее предупреждение до поломки №6 2018-07-08 было за 2 days 23:29:00
Последнее предупреждение до поломки №7 2018-07-25 было за 3 days 09:49:00
```

Рисунок 14. Показания датчика № 5 с выявленными аномалиями с помощью LOF и крайний временем предупреждения

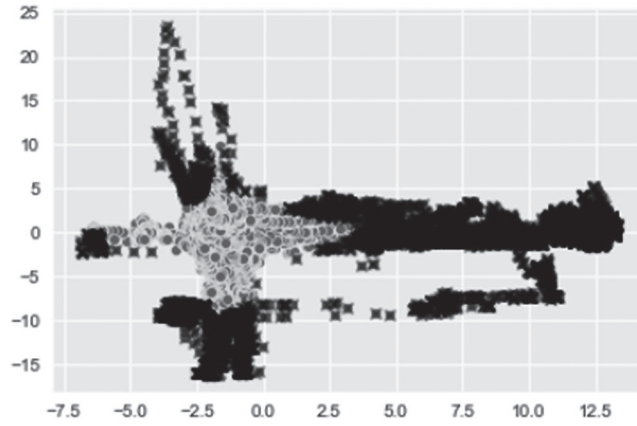


Рисунок 15. Детектирование аномалий с помощью Isolation Forest

Минимальным для параметра Contamination (загрязнения) [9], при котором алгоритм предупреждает все поломки, является значение 0,07. За весь период наблюдения выявлено 15640 аномалий.

Сравнительный анализ

В результате выполнения вычислительного эксперимента при обработке данных протестирована эффективность алгоритмов машинного обучения без учителя, основанных на плотности распределения точек (гауссовы смеси и локальный фактор выброса) и на деревьях решений (изолирующий лес). Каждый из них имеет гиперпараметры, которые настраиваются до обучения модели для получения оповещения перед каждой поломкой.


```
Последнее предупреждение до поломки №1 2018-04-12 было за 3 days 15:28:00
Последнее предупреждение до поломки №2 2018-04-18 было за 4 days 07:21:00
Последнее предупреждение до поломки №3 2018-05-19 было за 3 days 00:33:00
Последнее предупреждение до поломки №4 2018-05-25 было за 5 days 04:41:00
Последнее предупреждение до поломки №5 2018-06-28 было за 0 days 00:01:00
Последнее предупреждение до поломки №6 2018-07-08 было за 3 days 03:41:00
Последнее предупреждение до поломки №7 2018-07-25 было за 1 days 13:10:00
```

Рисунок 16. Показания датчика № 5 с выявленными аномалиями с помощью изолирующего леса и крайний временем предупреждения

Все обученные модели дали хорошие результаты, но с разным временем предупреждения об отказе насосной станции. Судя по графикам на Рисунках 12, 14, 16 можно сказать, что лучше всего проявляет себя LOF, так как максимальное время обнаружения поломки гораздо меньше, чем у других. Во внимание также нужно брать распределение аномалий вдоль всего временного ряда, так как частое ложное оповещение оператора на практике оказывается крайне нежелательным. В этом смысле гауссовы смеси дают лучший результат – аномальные точки перед поломкой сконцентрированы в узком промежутке времени. LOF обнаружил в 7 раз меньше аномалий, но они были разбросаны вдоль всего периода наблюдения. Изолирующий лес ложно определяет в качестве аномалий значительно большее количество данных (15640 аномальных точек), особенно перед третьей поломкой – предупреждения поступали около месяца.

В идеальном случае время предупреждения не должно быть либо слишком большим, либо очень маленьким, а именно таким, чтобы персонал смог оптимально и в сроки провести необходимое техническое обслуживание объекта, который действительно в скором времени может выйти из строя.

Заключение

Выявлено, что алгоритмы, основанные на плотности кластеров, лучше всего подходят к подобным данным [8]. За счет того, что данных достаточно много, изолирующий лес ложно детектирует большое количество точек как аномалии, что, в свою очередь, приводит к частым оповещениям. LOF имеет похожую проблему, но с меньшим числом аномалий. Не имеет данного недостатка Gaussian Mixture. Выбирая между ним и LOF, необходимо искать компромисс между точностью классификатора и длительностью интервала времени между определением аномалии и проявлением сбоя насосной станции.

Литература

1. Введение в анализ временных рядов: учебное пособие для вузов / Н.В. Артамонов, Е.А. Ивин, А.Н. Курбацкий, Д. Фантаццини; Московский государственный университет имени М.В. Ломоносова. Вологда: ВолНЦ РАН, 2021. 134 с. ISBN 978-5-93299-496-2.
2. *Мельников В.* Предобработка данных на Python [Электронный ресурс] / NTA. Data Science и AI в аудите. URL: https://newtechaudit.ru/predobrabotka-dannyh-na-python/ (дата обращения 15.07.2022)
3. *Aurélien G.* Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow: монография. O'Reilly, 2019. 483 с.
4. *Bauyrjan J.* Anomaly Detection in Time Series Sensor Data. Available at: https://towardsdatascience.com/anomaly-detection-in-time-series-sensor-data-86fd52e62538 (date of the application: 24.05.2022).
5. *Christopher B.* Time Series Analysis (TSA) in Python – Linear Modelsto GARCH. Available at: https://www.blackarbs.com/blog/time-series-analysis-in-python-linear-models-to-garch/11/1/2016 (date of the application: 03.09.2022).
6. *Jayaswal V.* Local Outlier Factor (LOF) – Algorithm for outlier identification. Available at: https://towardsdatascience.com/local-outlier-factor-lof-algorithm-for-outlier-identification-8efb887d9843 (date of the application: 18.09.2022).
7. *Kagumire S.* Anomaly Detection with Machine Learning. Available at: https://medium.com/mlearning-ai/anomaly-detection-with-machine-learning-8fa942fb5adc (date of the application: 30.07.2022).
8. Pump_sensor_data. Kaggle. URL: https://www.kaggle.com/datasets/nphantawee/pump-sensor-data (дата обращения: 07.06.2022)
9. sklearn.ensemble.IsolationForest. Scikit-learn Machine Learning in Python. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html (date of the application: 07.09.2022).
10. sklearn.mixture. Gaussian Mixture. Scikit-learn Machine Learning in Python. Available at:
11. https://scikit-learn.org/stable/modules/generated/sklearn.mixture. Gaussian Mixture. html#sklearn.mixture. Gaussian Mixture (date of the application: 24.08.2022).

References

1. Artamonov N.V., Ivin E.A., Kurbackij A.N., Fantaccini D. (2021) *Vvedenie v analiz vremennyh rjadov* [Introduction to Time Series Analysis]. Vologda, vol. NC RAN, 134 p. (in Russian). ISBN 978-5-93299-496-2.
2. *Mel'nikov V.* (2019) *Predobrabotka dannyh na Python* [Data preprocessing on Potkhon]. Available at: https://newtechaudit.ru/predobrabotka-dannyh-na-python/ (date of the application: 15.07.2022) (in Russian).
3. *Aurélien G.* (2019) Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow. O'Reilly, 483 p.
4. *Bauyrjan J.* Anomaly Detection in Time Series Sensor Data. Available at: https://towardsdatascience.com/anomaly-detection-in-time-series-sensor-data-86fd52e62538 (date of the application: 24.05.2022)
5. *Christopher B.* TimeSeriesAnalysis (TSA) inPython – Linear Modelsto GARCH. Available at: https://www.blackarbs.com/blog/time-series-analysis-in-python-linear-models-to-garch/11/1/2016 (date of the application: 03.09.2022).

6. *Jayaswal V*. Local Outlier Factor (LOF) - Algorithm for outlier identification. Available at: https://towardsdatascience.com/local-outlier-factor-lof-algorithm-for-outlier-identification-8efb887d9843 (date of the application: 18.09.2022)
7. *Kagumire S*. Anomaly Detection with Machine Learning / Medium.com. Available at: https://medium.com/mlearning-ai/anomaly-detection-with-machine-learning-8fa942fb5adc (date of the application: 30.07.2022)
8. Pump_sensor_data / Kaggle. Available at:
9. https://www.kaggle.com/datasets/nphantawee/pump-sensor-data (data obrashhenija 07.06.2022)
10. sklearn.ensemble.IsolationForest / Scikit-learn Machine Learning in Python. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html (date of the application: 07.09.2022)
11. sklearn.mixture.GaussianMixture / Scikit-learn Machine Learning in Python. Available at: https://scikit-learn.org/stable/modules/generated/sklearn.mixture. GaussianMixture.html#sklearn.mixture. GaussianMixture (date of the application: 24.08.2022).