Московский государственный университет
имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики

Отчет об участии в соревновании по анализу данных

«Sustainable Industry: Rinse Over Run»

Выполнил:
студент 203 группы
Федоров И. С.

Москва
2019

Содержание

Постановка задачи **2**

Подходы к решению задачи **5**

Прогнозирование временных рядов 5

Feature engineering 6

Финальный метод и его аргументация **6**

Описание всех сделанных экспериментов **8**

Возможные улучшения решения **9**

Результат **10**

Постановка задачи

Рассматривается процесс очистки загрязненного оборудования из области пищевой промышленности от частиц, бактерий, аллергенов и других потенциально опасных загрязнений с помощью системы CIP (Clean-In-Place). Особенность данного оборудования заключается в том, чтобы в процессе очистки объекты не разбираются на части.

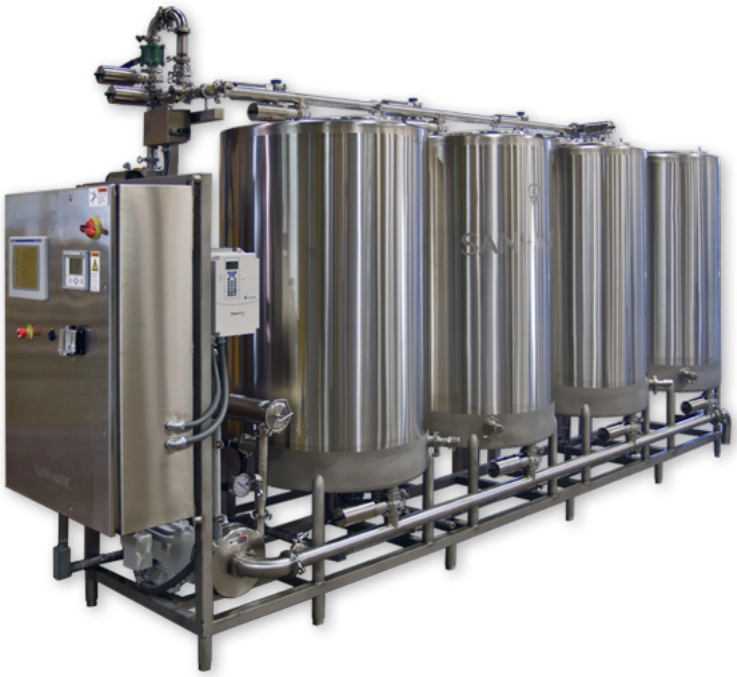

Рис. 1: Пример CIP системы

Очищаемые объекты проходят несколько этапов обработки, связанных с введением в них различных химических средств, а также промышленной воды.

Для контроля качества произведенной промывки оператору CIP станции предоставляется числовой показатель мутности. Он отображает меру загрязненности сточных вод, слитых из объекта на последнем шаге процесса очистки. Оценивая данную величину, оператор может принять решение о продлении этого этапа, если объект очищен недостаточно, либо сократить его в целях экономии ресурсов.

В течение каждого процесса система автоматически производит измерение некоторых величин каждые 2 секунды. К ним относятся, например, давление на линии подачи или концентрация кислоты в баке. Полный набор этих величин будет описан ниже. Целевая переменная (мутность) зависит от некоторых из них, полученных на последнем этапе.

Участникам предоставлено несколько наборов данных. Первым из них является обучающая выборка. В ней собраны данные, описывающие протекание различных процессов во времени. Таким образом, набор сведений о конкретном процессе очистки - это временной ряд. Обучающая выборка представлена в виде таблицы, в которой каждой строке соответствует набор данных, относящееся к одному из временных рядов. Условимся далее называть процессы объектами. Ниже представлены все доступные нам признаки, а также их описания.

Таблица 1: Метаданные

Признак	Описание
process_id	ID текущего процесса
object_id	ID объекта, над которым ведется процесс
phase	Этап процесса в данный момент
timestamp	Время
pipeline	Имя трубопровода

Таблица 2: Измерения

Признак	Описание
supply_flow	Поток жидкости, поступающей в трубопровод
supply_pressure	Давление в барах чистящих средств на линии подачи
return_temperature	Температура в C° чистящих средств на возвратной линии
return_conductivity	Проводимость в миллисименах чистящих средств на возвратной линии
return_turbidity	Мутность в единицах измерения чистящих средств на возвратной линии
return_flow	Поток в литрах в час чистящих средств на возвратной линии
supply_pump	Состояние (булево) насоса подачи на линии подачи
supply_pre_rinse	Состояние (булево) клапана предварительной промывки на линии подачи
supply_caustic	Состояние (булево) содового клапана на линии подачи
return_caustic	Состояние (булево) содового клапана на возвратной линии
supply_acid	Состояние (булево) кислотного клапана на линии подачи
return_acid	Состояние (булево) кислотного клапана на возвратной линии
supply_clean_water	Состояние (булево) клапана чистой воды на линии подачи
return_recovery_water	Состояние (булево) клапана возврата воды на возвратной линии
return_drain	Состояние (булево) сливного клапана на возвратной линии
object_low_level	Индикатор (булев) присутствия жидкости внутри очищаемого объекта (К примеру, жидкость останется, если насос не до конца очистит объект)
tank_level_pre_rinse	Уровень в процентах в баке предварительной промывки
tank_level_caustic	Уровень в процентах в содовом баке
tank_level_acid	Уровень в процентах в кислотном баке
tank_level_clean_water	Уровень в процентах в баке с чистой водой
tank_temperature_pre_rinse	Температура в C° в баке для воды
tank_temperature_caustic	Температура в C° в содовом баке
tank_temperature_acid	Температура в C° в кислотном баке
tank_concentration_caustic	Концентрация в миллисименсах в содовом баке
tank_concentration_acid	Концентрация в миллисименсах в кислотном баке
tank_lsh_caustic	Состояние (булево) переключателя высокого уровня заполненности в баке с содой (используется для определения, заполнен бак или нет)
tank_lsh_acid	Состояние (булево) переключателя высокого уровня заполненности в баке с кислотой (используется для определения, заполнен бак или нет)
tank_lsh_clean_water	Состояние (булево) переключателя высокого уровня заполненности в баке с чистой водой (используется для определения, заполнен бак или нет)
tank_lsh_pre_rinse	Состояние (булево) переключателя высокого уровня заполненности в баке для предварительной промывки (используется для определения, заполнен ли бак или нет)
target_time_period	Индикатор (булевый), учитывается ли данное наблюдение при подсчете целевой переменной

Как следует из таблицы 1, признак phase содержит в себе название текущего этапа очистки. Остановимся на нем чуть подробнее и рассмотрим всевозможные его значения. Всего существует 5 этапов:

- **pre_rinse** - промывочная вода загружена в объект
- **caustic** - сода загружена в объект
- **intermediate_rinse** - чистая или промывочная вода загружена в объект
- **acid** - азотная кислота загружена в объект
- **final_rinse** - чистая вода загружена в объект

Вторым доступным нам набором данных является таблица, содержащая значения целевой величины (мутности) для объектов (т.е. процессов) обучающей выборки. В некотором смысле она является избыточной, поскольку мы можем самостоятельно посчитать эту величину по обучающей выборке. Для n-го процесса значение t_n мутности можно найти по формуле

$$t_n = \langle \mathbf{x}, \mathbf{y} \rangle$$

- $\langle \mathbf{a}, \mathbf{b} \rangle$ - скалярное произведение векторов $\mathbf{a}$ и $\mathbf{b}$
- $\mathbf{x}$, $\mathbf{y}$ - вектора, составленные из значений измерений $\max(\text{return\_flow}, 0)$ и return_turbidity соответственно, для которых значение признака target_time_period истинно

Заметим, что булевый признак target_time_period принимает истинное значение только на последнем этапе очистки (final_rinse).

Последним из основных наборов данных является тестовая выборка, которая по своей структуре полностью совпадает с обучающей выборкой. Однако между ними существует ключевое различие. Дело в том, что ни для одного объекта из тестовой выборки нам не даны сведения о финальной фазе. Более того, чаще всего для этих процессов мы наблюдаем отсутствует сразу нескольких фаз. По информации организаторов, временные ряды обрываются на одном из этапов:

- Для 10% процессов, данные обрываются на 1 этапе.
- Для 30% процессов, данные обрываются на 2 этапе.
- Для 30% процессов, данные обрываются на 3 этапе.
- Для 30% процессов, данные обрываются на 4 этапе.

В качестве дополнительного источника информации, организаторы предоставили таблицу, содержащую планы ("рецепты") очистки для каждого из процессов обучающей и тестовой выборки. Под планом подразумевается набор этапов, который должен пройти объект во время соответствующего процесса. Сразу заметим, что эти данные не всегда совпадают с действительностью. В исследовании, приложенному к данному отчету, показывается, что для обучающей выборки мы находим около 5% ошибок в этом дополнительном наборе планов.

Рассмотрев все предоставленные нам наборы данных, вернемся к постановке задачи. Участникам предлагается проанализировать обучающую выборку, учитывая известные для каждого объекта этой выборки значения целевой величины. С помощью полученной информации построить некоторую математическую модель и, используя её, спрогнозировать значение мутности для объектов тестовой выборки. При этом могут использоваться дополнительные данные из набора рецептов для каждого процесса. Использовать какие-либо иные источники информации для решения этой задачи запрещено по правилам соревнования.

Для оценки качества полученного предсказания используется модифицированная функция ошибки MAPE (mean absolute percentage error). Её значение вычисляется по формуле

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{y}_i - y_i|}{max(|y_i|, 290000)}$$

- N - число предсказанных значений
- $\hat{y}_i$ - предсказанное значение мутности
- y_i - истинное значение мутности

Таким образом, целью соревнования является получение набора предсказаний, для которых значение ошибки MAPE минимально. На основании её значения выстраивается рейтинг участников.

Подходы к решению задачи

В данном отчете будут рассматриваться лишь подходы, использующие статистические методы и модели машинного обучения.

Прогнозирование временных рядов

Поскольку основными объектами, с которыми мы имеем дело в этой задаче, являются временные ряды, то логично предложить решение задачи с помощью прогнозирования временных рядов, то есть предсказания некоторого количества значений, которые примет измеряемая величина после обрыва данных нам сведений. Например, можно спрогнозировать интересующие нас измерения return_flow и return_turbidity на финальной фазе для каждого объекта тестовой выборки, и посчитать по описанной выше формуле целевую величину.

Существует множество статистических моделей для решения задачи прогнозирования временного ряда. Например, можно попробовать использовать модель авторегрессии - скользящего среднего ARIMA. Однако, при данном подходе, мы столкнемся с некоторыми трудностями.

Для начала, если мы посмотрим на графики, представленные в исследовании, то увидим, что временные ряды не являются независимыми, а графики важных нам измерений представляют скорее случайный шум, чем осмысленную комбинацию тренда, сезонности и циклов. Таким образом, для построения подобных авторегрессионных моделей нам нужно учитывать значения всех признаков сразу.

Кроме того, длина прогнозируемого участка для большинства рядов тестовой выборки значительно превышает доступную нам часть, что понижает точность, так как на последней фазе ряд зачастую будет прогнозироваться исходя из предыдущих значений, которые также были спрогнозированы алгоритмом. Вследствие этого ошибка будет постоянно накапливаться.

Заключительной проблемой, с которой мы столкнемся, используя данный подход, будет тот факт, что не все значения нужных нам признаков return_flow и return_turbidity на последней фазе учитываются при подсчете мутности, а лишь те, для которых значение булевого признака target_time_period истинно. Можно строить различные гипотезы, при каких условиях данные входят в финальное выражение, но это все лишь увеличивает ошибку.

Feature engineering

Другим подходом к решению данной задачи является метод генерации разнообразных признаков на основании значений показателей имеющихся временных рядов. К подобным признакам можно отнести различные статистические характеристики, такие как среднее значение, медиана, среднеквадратичное отклонение, дисперсия, максимум, минимум и так далее. При использовании данного метода для каждого временного ряда создается единственный вектор признаков. В конце из таких наборов данных составляется матрица объект - признак. В контексте нашей задачи в качестве объектов будут выступать процессы, а в качестве признаков их статистические характеристики.

Составив такую матрицу, мы сводим исходную задачу к классической задаче машинного обучения - регрессии (поскольку целевая величина является вещественным числом). Для решения задачи регрессии существует обширное множество известных алгоритмов. Например, линейные алгоритмы, логистическая регрессия или случайный лес.

Заметим, что при использовании данного метода, мы учитываем сразу все признаки, а не считаем их независимыми, как при попытке спрогнозировать заведомо зависимые временные ряды по отдельности.

Рассмотрим теперь проблемы, с которыми мы стокнемся, если захотим использовать данный подход.

Ключевой проблемой, возникающей при генерации признаков, является подбор их таким образом, чтобы они максимально информативно отображали особенности временного ряда. Ведь по сути мы пытаемся сильно понизить размерность пространства объект - признак. Понятно, что при применении данного метода к достаточно сложному по структуре временному ряду часть информации будет утеряна. Нашей задачей является минимизация этой потери.

Следующей проблемой, относящейся к конкретно нашей задаче, является тот факт, что данные представлены не в полном объеме: в большинстве случаев мы, помимо финальной фазы, не имеем данных о некоторых промежуточных фазах. Так, при построении признаков, мы должны некоторым образом учитывать эти пробелы в данных, прогнозируя (в обобщенном смысле) значения признаков, как если бы этих промежутков не существовало.

Финальный метод и его аргументация

Оценивая преимущества и недостатки описанных выше подходов применительно к данной задаче, я принял решение использовать feature engineering. Инструментарием для создания решения служил язык Python 3.7 и его библиотеки pandas, numpy, sklearn и matplotlib.

В качестве основы для решения я взял предоставленный организаторами пример, который далее я буду называть бенчмарком (http://drivendata.co/blog/rinse-over-run-benchmark).

Публичное решение группирует временные ряды по процессу и для каждой такой группы создает два логически разных множества признаков. Стоит отметить, что процессы, для которых известна только финальная фаза, сразу отбрасываются, ведь они не несут в себе никакой важной информации. Единственным применением таких данных могло бы служить прогнозирование ряда "назад во времени однако это было запрещено организаторами.

К первой группе относятся признаки, относящиеся к метаданным. Как мною было установлено в исследовании, на протяжении каждого конкретного процесса имя трубопровода (pipeline) не меняется. Эта идея отражается в публичном решении: для каждого процесса создаются так называемые dummy признаки - небольшой набор бинарных признаков, каждый из которых соответствует своему имени трубопровода. Разбиение единого категориального признака на несколько бинарных может иметь смысл, поскольку в дальнейшем

используется модель случайного леса. Если бы признак был единый (к примеру, значение 1 соответствовало бы первому трубопроводу, 2 - второму и т.д.), то для случайного леса мог бы неявно возникнуть порядок на данном множестве, хотя на самом деле его нет. Вторым (и последним) метапризнаком, предлагаемым публичным решением является число фаз, которое прошел объект во время очистки. Вполне логичный признак, ничего необычного.

Ко второй группе признаков относятся статистические характеристики некоторых измерений, проводимых во время процесса, а именно берутся все измерения, не являющиеся бинарными. Для каждой из этих величин генерируется 5 характеристик:

- max - максимум
- min - минимум
- mean - среднее значение
- std - среднеквадратическое отклонение
- tail mean - среднее значение за последние 5 измерений

Важной особеностью публичного решения является то, что для обучения используется только 80% выборки. Было установлено, что если брать 100% данных, то результат будет хуже. Обоснование этому и развитие этой идеи найдет место в моем решении.

После всех предварительных работ запускается алгоритм случайного леса (из библиотеки sklearn), который прогнозирует целевую величину. Единственным отличием настроек алгоритма от стандартных является измененное число решающих деревьев (n_estimators = 1000).

Обратим внимание на бросающиеся в глаза недостатки данного решения. Первой и главной проблемой является подсчет данных характеристик на протяжении всего процесса. Очевидно, что потеря информативности будет просто колоссальной, особенно учитывая тот факт, что процессы представлены разным количеством фаз. Во-вторых, параметры случайного леса выбраны практически стандартными, что требует доработки, поскольку неверные параметры могут значительно ухудшить качество прогноза. В-третьих, часть выборки отбрасывается случайным образом, что вносит слишком высокую долю случайности в работу. Также к недостаткам стоит отнести достаточно неточные признаки, особенно tail mean - ведь каждый процесс тестовой выборки может закончиться на любой фазе, поэтому совсем некорректно сравнивать их средние значения в моменты последних измерений.

Мое решение основано на исправлении недостатков бенчмарка, а также развитии идеи с ограничением обучающей выборки.

Для начала я рассмотрел метапризнак object_id. В моем исследовании было установлено, что, как и в случае с pipeline, на протяжении всего процесса значение данного признака остается постоянным. То есть очищаемый объект не меняется во время очистки. Более того, было установлено, что все объекты, проходящие очистку в тестовой выборке, уже проходили её в обучающей. Можно предположить, что объекты могут делиться на различные категории (например, вилки и тарелки), и для каждой из них существует своя особенность очистки. В связи с этим, я решил добавить данный признак к составляемому набору данных.

Для понижения потери информативности я поступил самым логичным (по моему мнению) способом - сгруппировал временные ряды не только по процессам, но и по фазам. Таким образом, в моем решении статистические признаки генерируются не для всего процесса сразу, а для каждого его этапа очистки. Поскольку для многих процессов некоторые фазы отсутствуют, то на выходе я получал промежутки с неопределенным значением NaN. Так как для выполнения задания было достаточно мало времени, я поступил самым примитивным образом: заполнил их медианами по соответствующим признакам. Возможные методы решения этой проблемы я опишу в дальнейших разделах отчета.

Генерируемые признаки также были изменены и дополнены. При работе с временными рядами часто используются так называемые rolling windows. Это окна некоторого фиксированного размера, которые двигаются по ряду с некоторым шагом, а на каждом шаге вычисляется некоторая характеристика этого окна (например, среднее значение). Поскольку в данной задаче проводится довольно большое количество измерений, то считать эти окна для каждого признака слишком часто невыгодно. В моем решении я сильно упрощаю эту идею, оставив всего несколько окон: в начале фазе, в середине и в конце. Для каждого из них в качестве признака считается среднее значение. Кроме того, характеристики max и min в моем решении заменены на единый признак difference = max - min. Поскольку модели будет известно приблизительное значение ряда в начале, середине и конце, то мы не потеряем информативности, а размерность понизится.

В качестве модели прогнозирования использовался случайный лес, как и в бенчмарке, однако его параметры были улучшены. А именно была ограничена максимальная глубина max_depth = 25 и установлено число просматриваемых при расщеплении деревьев случайных признаков max_features = 100. Глубина была подобрана экспериментально, а число признаков - по рекомендации из блога А. Г. Дьяконова $\frac{n}{3}$ для задачи регрессии, где n - число признаков в обучающей выборке (в моем решении $n \approx 300$). Значение n_estimators осталось равным 1000.

Отправив несколько достаточно успешных предсказаний, было принято решение вручную посмотреть оценить влияние признаков на предсказание. Эти показатели предоставляет случайный лес (feature_importances_). Было установлено, что признаки, генерируемые из измерений tank_concentration_acid и tank_concentration_caustic вносят пренебрежимо малый вклад. В связи с этим было принято решение исключить их из модели, в целях борьбы с переобучением.

Причина того, что на урезанной выборке качество прогноза улучшалось, может заключаться в особенностях функции ошибок, используемой в соревновании. Посмотрим на неё ещё раз.

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|\hat{y}_i - y_i|}{max(|y_i|, 290000)}$$

Несмотря на то что организаторы модифицировали MAPE и добавили в знаменатель выбор максимума из $|y_i|$ и 290000, значение $|y_i|$ может быть достаточно близко к этому пределу. Таким образом, значение ошибки зависит не только от качества предсказания, но и от масштаба прогнозируемой величины. Чем больше её значение, то меньше ошибка (при равном модуле разности истинного значения и предсказанного). В связи с этим экспериментальным путем было установлено, что при более точном обучении на процессах, имеющие небольшой показатель мутности, итоговое предсказание получается качественнее. Это может происходить из-за переобучения: если в выборке присутствуют слишком большие значения, то модель может под них подстроиться, принебрегая ошибкой на маленьких по модулю величинах. В данном случае этот обмен будет неравносильный. Опять же, экспериментальным путем на кросс-валидации и отправках предсказаний на сайт, было установлено, что оптимальным решением будет отбросить ту часть выборки, для которой значений целевой величины превосходит 900000.

Описание всех сделанных экспериментов

Велось достаточно подробное протоколирование всех идей, которые мне приходили во время выполнения задания, а также мои рассуждения. Вся эта информация доступна в прилагаемом текстовом файле (никаких формальностей, просто мысли вслух). Изучив данный

файл, можно видеть, что принципиально других моделей я не использовал.

Практически сразу пришел к идее с разбиением на фазы. В качестве обучаемой модели машинного обучения пробовал CatBoost, XGBoost, KNN. По результатам экспериментов, все они уступали случайному лесу, хотя, возможно, я просто недостаточно корректно их настраивал.

Некоторое время я пытался максимально эффективно использовать дополнительные данные, то есть план очистки для каждого объекта. И обучающая, и тестовая выборки разбивались на паттерны (то есть планы очистки), и при прогнозировании значения для объекта из некоторого паттерна, учитывалась только та часть обучения, которая ему соответствует. Однако в конце я отказался от этой идеи, поскольку, во-первых, планы содержат около 5% ошибок (обоснование см. в постановке задачи), а во-вторых, для тестовых объектов отсутствует слишком много данных, чтобы реализовать эту идею. Приходится дозаполнять невероятно большие куски информации, что ведет к серьезным ошибкам.

Большинство экспериментов вносили в решение небольшие изменения. Например, разные ограничения на выходной параметр для обучающей выборки, подбор оптимальных параметров случайного леса и так далее.

Возможные улучшения решения

У предложенной мною модели есть несколько недостатков.

Первым из них является заполнение отсутствующих данных крайне грубым путем - подстановкой медианного значения по соответствующему столбцу. Поскольку зачастую приходится выполнять эту операцию для больших объемов данных, то качество прогнозирования сильно ухудшается. Решением этой проблемы я вижу в умном заполнении промежутков NaN, а именно построении промежуточных моделей для конкретных признаков, которые обучаются на доступной (не NaN) части выборок (как обучающей, так и тестовой), и прогнозируют оптимальное значение недостающих признаков. Стоит также задуматься о том, чтобы количество таких моделей было как можно более меньшим, поскольку отсутствующих признаков могут быть десятки и сотни.

Вторым из них является генерация довольно грубых признаков. Несмотря на то что высчитывается достаточно много статистических характеристик, они несут в себе не очень много полезной информации. Возможно, нужно вернуться к идеям авторегрессии и скользящего среднего, и некоторым образом добавить их в качестве признаков. Организовать поиск во временном ряде (в т.ч. многомерном) наиболее информативных признаков может помочь библиотека tsfresh (https://github.com/blue-yonder/tsfresh). Данная библиотека автоматически анализирует ряд и генерирует (в стандартном случае) более 700 различных признаков. Кроме того, она, учитывая целевую величину, предоставляет возможность отбора наиболее информативных из них. За короткий срок я не смог достаточно глубоко разобраться в данной библиотеке, чтобы эффективно и правильно её использовать. Дело в том, поскольку у нас имеется достаточно большое число измерений, количество генерируемых признаков требует гораздо больших вычислительных возможностей, чем у меня имеется, поэтому я просто физически не смог её воспользоваться. Скорей всего, нужно некоторым образом группировать временные ряды, чтобы снизить суммарные вычислительные требования, то есть выполнять работу по частям.

Последней по приоритету идеей является возврат к прогнозированию. Наверняка существуют различные модификации моделей семейства ARIMA для работы с многомерным временными рядами.

Результат

Финальный результат: **0.4999**
Итоговое место: **84**
Число посылок: **11**
Ник: **Sorrow**
Аккаунт: **https://www.drivendata.org/users/Sorrow/**

Таблица 3: Результаты посылок

№ посылки	Публичный результат	Приватный результат
1	1.4636	1.6004
2	1.4791	1.6237
3	1.4962	1.6363
4	1.4071	1.3548
5	1.3649	1.2666
6	3.7090	4.2156
7	2.2150	2.2542
8	1.5879	1.4988
9	0.5212	0.5256
10	0.5160	0.5202
11	0.4994	0.4999