Мониторинг сдвига данных в моделях машинного обучения

Д.Е. Намиот, Е.А. Ильюшин

***Аннотация*— Принципиальным моментом эксплуатации систем машинного обучения является то, что модели обучаются на некотором выделенном тренировочном наборе данных. Соответственно, обобщения, полученные на этапе тренировки, обусловлены характеристиками некоторого подмножества генеральной совокупности. Если характеристики данных меняются на этапе эксплуатации системы, то обобщения модели становятся, вообще говоря, несостоятельными. При этом, такое изменение данных следует считать скорее правилом, чем исключением. Такое изменение характеристик данных называется сдвигом данных. Это, в свою очередь означает, что любая система машинного обучения, претендующая на роль промышленной, должна отслеживать возможный сдвиг данных. Наличие такого сдвига снижает доверие к результатам работы или даже делает систему непригодной для дальнейшей эксплуатации. Учет (преодоление) такого сдвига данных – это отдельная задача, простое переобучение может оказаться большой проблемой для критических приложений, например. Но в любом случае, первая задача – это определение факта сдвига данных. Сам сдвиг данных подразделяется на несколько типов, самым серьезным из которых является изменение связи между зависимыми и независимыми переменными. Естественно, что особый интерес вызывает определение сдвига данных для потоков, поскольку это непосредственно связано с критическими приложениями.**

***Ключевые слова*— машинное обучение, мониторинг, сдвиг данных.**

I. Введение

Работа моделей машинного обучения (неожиданно!) зависит от данных. На самом деле, например, темы, связанные с так называемым feature engineering (то есть выбором данных для систем машинного обучения) [1] являются определяющими для последующей работы таких систем. Именно специальные воздействия на данные (специальные модификации данных) на этапе тренировки или исполнения моделей и называются атаками на системы машинного обучения [2]. Такие атаки (точнее – невозможность им противостоять) и является главной причиной проблем внедрения систем машинного обучения в критические приложения (авионика, автоматическое вождение и т.п.) [3]. И в целом, цифровая экономика – это наука о данных [4].

Статья получена 9 августа 2022.
Д.Е. Намиот - МГУ имени М.В. Ломоносова (email: dnamiot@gmail.com).
Е.А. Ильюшин - МГУ имени М.В. Ломоносова (email: john.ilyushin@gmail.com)

При этом неоднократно отмечалось, что и без специального воздействия на данные, работающие системы машинного обучения, показывавшие допустимые метрики на тренировочных и тестовых данных оказываются чувствительны даже к небольшим вариациям исходных данных и обобщения, достигнутые на этапе тренировки модели, не работают при реальной эксплуатации системы. В целом, это общая проблема для всех дискриминантных моделей машинного обучения - реальные данные могут отличаться от данных, на которых модель тренировали и тестировали. Более того, во многих случаях это следует рассматривать, скорее, как правило. Генеральная совокупность данных остается неизвестной, реальные данные могут сколь угодно сильно отличаться от данных, использованных на этапе тренировки. Это и означает, что обобщения модели, полученные на этапе тренировки, могут не работать, полученные и подтвержденные тестированием метрики не выполняться. Это, в свою очередь, означает, что во время работы (промышленной эксплуатации) модели машинного обучения необходимо следить за соответствием характеристик реальных данных характеристикам тренировочных данных. Это один из главных моментов в мониторинге работающих моделей машинного обучения. Определение так называемого сдвига данных необходимо для определения момента снижения достоверности результатов работы модели, или даже определения факта невозможности ее дальнейшего использования без обновления. При этом сам сдвиг данных неоднороден по своему воздействию и характеристикам проявления. Рассмотрению анализа сдвига данных и посвящена настоящая статья.

Оставшаяся часть статьи структурирована следующим образом. В разделе II рассматриваются причины и последствия сдвига данных. В разделе III приводятся формальные определения сдвига данных и возможные характеристики. Раздел IV посвящен инструментам измерения. И в разделе V, играющем роль заключения, мы представляем свою архитектуру для системы мониторинга данных.

II. Причины появления и последствия сдвига данных

Фундаментальное предположение при разработке любой модели машинного обучения заключается в том, что тренировочные данные, которые используются для обучения модели, точно представляют всю генеральную

совокупность данных. Исходя из этого предположения, мы считаем, что обобщения, достигнутые на этапе тренировки, сохранятся при реальном использовании. Но это, на самом деле, лишь предположение. И проблема в том, что об отличии реальных данных от тренировочных (и тестовых), мы сможем узнать уже только после запуска модели (системы, использующей машинное обучение) в производство (в реальную работу). Почему это важно? Различие в данных может привести к тому, что обобщения, сделанные на этапе тренировки модели, больше не работают. Особенно остро эта проблема стоит для так называемых критических применений (авионика, автоматическое вождение и т.д.). Такого рода системы запускаются в промышленную эксплуатацию только после получения на этапе тренировки (тестирования) допустимых показателей (метрик) работы. И очевидно, что устойчивость (безопасность) для таких систем – это сохранение таких показателей во время работы (то есть – на всех данных). Объяснение этому самое простое – эксплуатация критических систем вне заявленных параметров работы должна быть просто невозможной.

Когда вы обучаете модель в случае обучения с учителем, данные обучения обычно помечаются. Когда вы начинаете использовать модель на реальных данных, фактической метки нет. И независимо от того, насколько точна была ваша модель на этапе тренировки (тестирования), прогнозы верны только в том случае, если реальные данные, представленные на этапе эксплуатации, имитируют (или статистически эквивалентны) данные, использованные при обучении. То есть наш тренировочный набор данных являлся надежным представлением генеральной совокупности, которая, в общем случае, неизвестна (по крайней мере – неизвестна на момент тренировки модели). Что, если это не так? Отступления от этой эквивалентности и называются дрейфом (сдвигом) данных. Типичный пример изображен на рисунке 1.

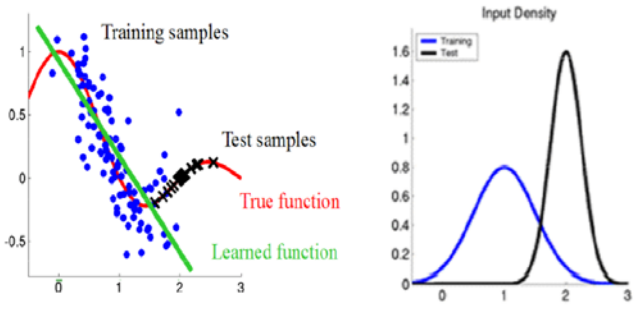

Рис. 1. Ковариационный сдвиг данных

Здесь рассматривается задача аппроксимации неизвестной функции (красная линия). Тренировочные данные (синие точки) распределены по нормальному закону со средним 1. Модель хорошо обобщается как линейная регрессия (зеленая линия). Но если мы выберем в качестве тестовых данных такое же нормальное распределение, но со средним 2 (черные точки), то ошибка будет только расти с ростом аргумента. Распределение реальных данных сдвинуто (правый рисунок) по сравнению с тренировочным набором.

Дрейф данных определяется как отклонение реальных данных (данных на этапе использования) от данных, которые использовались для тестирования и проверки модели перед ее развертыванием в производственной среде. Есть много факторов, которые могут привести к дрейфу данных. Одним из ключевых факторов является просто изменение времени. Если мы обратимся к типичному конвейеру системы машинного обучения (рис. 2), на котором показаны высокоуровневые этапы разработки модели машинного обучения, станет очевидным, что существует значительный разрыв между временем сбора тренировочных данных и использованием модели, работающей на реальных данных.

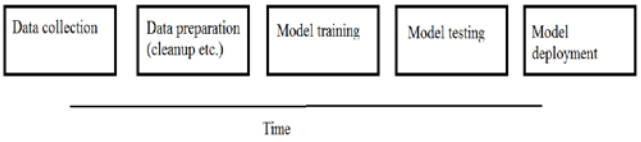

Рис. 2 Конвейер машинного обучения во времени

Этот разрыв может варьироваться от недель до месяцев и лет, в зависимости от сложности проблемы. Ведь этапы тренировки и тестирования являются итеративными. Сколько примеров моделей было разработано на давно собранных датасетах? Очевидно, что не для всех доменов (предметных областей) характеристики данных остаются постоянными.

Другие факторы, которые могут вызывать дрейф - это, например ошибки при сборе данных, сезонность, качественные изменения в окружении (классические примеры – предсказание посещения ресторанов до и во время COVID-19, когда данные собирались до пандемии, а модель развертывается после пандемии), простое непонимание всех аспектов работы моделируемой системы, зависимости в данных.

Что может быть, если дрейф данных никак не идентифицируется? В общем случае, это снижает нашу уверенность в достоверности результатов работы модели. Хуже всего то, что мы вообще не будем знать о снижении такой уверенности. Естественно, что результаты возможного нарушения обобщений этапа тренировки будут различными, в зависимости от бизнес-приложений. Неправильные рекомендации всегда будут менее опасны, чем неправильные действия.

Что делать, если дрейф обнаружен? Самый правильный ответ заключается в том, что нет общих (универсальных) рекомендаций. В каких-то случаях дрейф данных не будет оказывать влияния на результат. Есть понятие сертифицированной робастности, которое описывает устойчивость собственно модели.

Естественно, что во многих работах упоминают перетренировку модели. Здесь нужно отметить несколько моментов. Во-первых, перетренировка модели занимает какое-то время. Как будет работать наша система в течение этого времени? Что если мы имеем дело с критическим приложением, работающим 24x7? Следующий вопрос – а сколько раз придется проводить этот процесс? На настоящий момент можно сказать, что применяемые модели машинного обучения в целом ориентированы на неизменность характеристик данных генеральной совокупности относительно тренировочного набора.

Есть еще подход, связанный с онлайн-обучением, то есть переобучением при каждом наблюдении. Но здесь возникает проблема состязательных атак, которые при таком подходе труднее обнаруживать и базовая проблема с критическими сервисами – мы не можем эксплуатировать систему, метрики которой ниже некоторого жестко установленного порога.

Нижеследующие определения являются стандартными в литературе, посвященной анализу сдвига данных. Введем следующие формальные определения:

X - пространство данных,

Y - множество меток данных

P(X): распределение данных

P(Y): распределение меток

P(X|Y): распределение данных с заданными метками

P(Y|X): распределение меток с заданными данными

P (X,Y): совместное распределение. Фактически – это то, что и изучает модель машинного обучения

В этом случае различные дрейфы данных определяются так:

- Ковариационный сдвиг
  P(Y|X) остается неизменным, а P(X) - изменяется. На рисунке 2 как раз и изменялось распределение входных данных. Это сдвиг в независимых переменных.
- Сдвиг меток (сдвиг априорной вероятности или зависимой переменной)
  P(Y) изменяется, а P(X|Y) - неизменно
- Сдвиг концепции
  P(Y|X) изменяется, а P(X) - неизменно
  В некоторых работах говорят о совместном распределении. Концептуальный сдвиг – это сдвиг в отношениях (связях) между независимыми и зависимыми переменными: $P_t(X,Y)!=P_{t1}(X,Y)$, то есть совместное распределение меняется со временем.

Это проиллюстрировано на следующем рисунке

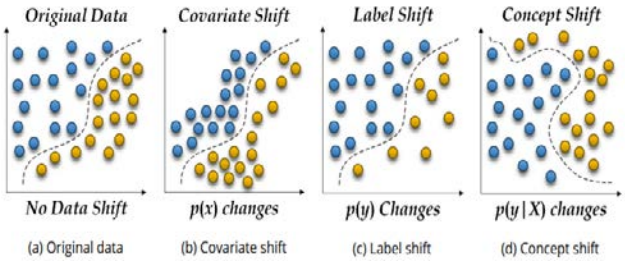

Рис. 3. Сдвиги данных [6]

Сдвиг концепций часто еще различают по типу (характеру) изменений. Например, постепенный дрейф концепции происходит результате динамичного характера бизнеса-модели (предметной области). Например, обнаружение мошенничества в финансовых транзакциях является одной из многих областей, где это может произойти. Поскольку мошенники постоянно адаптируются к методам обнаружения мошенничества, модель, основанная на исторических данных о мошеннических транзакциях, становится неэффективной. Повторяемость шаблонов данных, характерных для мошеннических сделок падает. Это также может произойти, когда пользователи или потенциальные пользователи продукта меняют свои предпочтения со временем.

Повторяющийся концептуальный дрейф обусловлен периодическими сезонными событиями. Это может произойти, когда, например, поведение клиентов меняется с сезонными изменениями. Это может произойти, когда есть скидки на продукты и предпочтения пользователей в течение определенного периода времени. Это то, что называется «сезонностью», и это, в принципе, может быть предсказано до начала разработки, но сама модель ML не учитывает такой фактор.

И, наконец, выделяют мгновенный дрейф концепции. Обычно это вызвано внезапными событиями, которые могут рассматриваться как аномалии по сравнению с обычными наблюдениями. Технически, это может произойти и из-за проблем с качеством данных (качеством измерений). При этом аномалии могут оказаться и довольно длительными по времени. Классический пример последнего времени – это пандемия COVID-19. Очевидно, что такое событие никогда не планировали. При этом пользователей резко изменилось и неравномерно повлияло на бизнес. Это исключительное событие, и оно, естественно, затронуло и модели, построенные на основе исторических данных.

Концептуальный сдвиг может быть виртуальным, как показано в работе [7] (рис. 4).

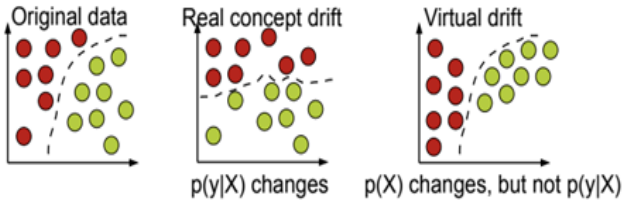

Рис. 4. Виртуальный и реальный сдвиг концепции [7]

III. Каким образом можно отслеживать дрейф данных?

Дрейфы данных можно выявить с помощью методов последовательного анализа, методов на основе моделей и методов на основе оценки распределения. Методы последовательного анализа, такие как DDM (метод обнаружения дрейфа)/EDDM (ранний DDM), полагаются на частоту ошибок для определения обнаружения дрейфа, метод на основе моделей использует пользовательскую модель для определения дрейфа, а методы на основе временного распределения используют вычисления расстояния между распределениями. Некоторыми из популярных статистических методов для расчета разницы между любыми двумя популяциями являются Population Stability Index, Kullback-Leiber or KL Divergence или KL, расхождение Дженсона-Шеннона или JS, тест Колмогорова-Смирнова или тест KS, метрика Вассерштейна.

DDM базируется на идее последовательной оценки ошибки классификации новых экземпляров данных. Когда появляется новый экземпляр данных, он классифицируется по текущей модели. При этом система отслеживает последовательность ошибок классификации. Когда ошибка превышает некоторый заданный предел – это и является признаком изменения в распределении данных.

Этот подход показывает хорошие результаты, когда мы имеем дело с резкими или не очень медленными изменениями. Но если изменения происходят медленно, то детектирование может занять довольно долгое время. Иными словами, система поздно определит сдвиг данных.

Метод раннего обнаружения дрейфа (EDDM) был разработан для улучшения обнаружения в условиях постепенного дрейфа концепции. В то же время он сохраняет хорошую производительность в ситуациях с резким дрейфом концепции. Основная идея заключается в рассмотрении расстояния между классификацией двух ошибок, а не только количества ошибок. Уменьшение такого расстояния свидетельствует об отсутствии прогресса в обучении, который и связан с изменением данных.

Очевидно, что оба этих подхода основаны на сборе данных в течение некоторого времени. В частности, базовая работа по EDDM говорит о 30 отсчетах [8].

Метод на основе моделей предлагает использовать модели специальные модели машинного обучения для определения сдвига данных.

Индекс стабильности населения (PSI) сравнивает распределение оценочной переменной (прогнозируемой вероятности) в наборе оценочных данных с обучающим набором данных, который использовался для разработки модели. Идея состоит в том, чтобы проверить, «как текущая оценка сравнивается с прогнозируемой вероятностью из набора обучающих данных».

A – оцениваемый набор данных
B – тренировочный набор

PSI = (% записей с проверяемой переменной в A - % записей с проверяемой переменной в B) * ln(A/ B)

Критерии оценки:

PSI < 0.1 – Нет изменений. Можно продолжать использовать существующую модель.

PSI >=0.1 & PSI<0.2 - Требуются небольшие изменения.

PSI>=0.2 – Требуются значительные изменения. Вообще говоря, текущая модель не должна больше использоваться.

Дивергенция Кульбака-Лейблера (или KL) - это мера того, насколько одно распределение вероятностей отличается от второго, эталонного распределения вероятностей.

$$D_{KL}(P \parallel Q) = \int_{-\infty}^{+\infty} p(x)\log(\frac{p(x)}{q(x)})dx$$

Дивергенция Дженсена-Шеннона (или JS) — это метод измерения сходства между двумя распределениями вероятностей. Он основан на KL-дивергенции, но симметричен и всегда имеет конечное значение.

$$JSD(P \parallel Q) = \frac{1}{2}D(P \parallel M) + \frac{1}{2}D(Q \parallel M)$$

где M= $\frac{1}{2}(P+Q)$

Для числовых признаков мы можем использовать двухвыборочный критерий Колмогорова-Смирнова. Проверяется, принадлежат ли тестовые и обучающие данные одному и тому же распределению или имеется статистически значимое различие. В этом случае статистика Колмогорова–Смирнова равна

$$D_{n,m} = \sup_x \left|F_{1,n}(x) - F_{2,m}(x)\right|$$

где $F_{1,n}$ и $F_{2,m}$ эмпирические функции распределения первой и второй выборок соответственно. Для больших выборок, где n и m — размеры первой и второй выборок соответственно, нулевая гипотеза отклоняется на уровне α, если

$$D_{n,m} > c(\alpha)\sqrt{\frac{n+m}{n\times m}}$$

Для бинарных категориальных признаков мы можем использовать простой Z-тест на разницу в пропорциях. Как часто обучающие и тестовые данные имеют одно из двух значений бинарного признака? Этот тест должен проверить, есть ли статистически значимая разница.

А для многомерных категориальных признаков мы можем использовать критерий хи-квадрат. Этот тест направлен на то, чтобы увидеть, вероятно ли распределение функции в тестовых (реальных) данных основано на распределении в обучающих данных.

При этом необходимо отметить следующее. При всей простоте и ясности, статистические тесты оказываются достаточно ресурсоемкими. Необходимо ведь учитывать количество характеристик и временной интервал, на котором они будут собираться. Например, в [13] указывается, что статистические тесты на потоке Twitter и временном окне 10 минут при тысячах характеристик занимает 12 секунд. Такое время оценки сдвига не подходит для высокоскоростных потоков.

Отметим также, что при анализе сдвига концепций использованные методы могут различать по тому, есть у нас истинные метки для новых данных или нет. Если метки доступны, то сигналом о сдвиге концепций может служить точность классификатора. Она должна снижаться по сравнению со значениями, полученными на этапе тренировки (тестирования). Если таковых меток нет, то это другой класс алгоритмов – labelless methods. Мы можем интересоваться распределением характеристик (features) новых данных. В работе [9] предлагается другой подход к такому анализу, когда вместо характеристик данных используются объясняющие переменные Shapley [10]. Сдвиг концепций определяется по сдвигу объясняющих переменных. Отметим, что такой подход одновременно позволяет понять, где именно произошел сдвиг.

Отдельно хотелось бы выделить непараметрические методы оценки смещения, работающие в реальном времени. При этом из них выделить методы, которые не используют информацию о разметке реальных данных. Именно такая форма представления задачи мониторинга сдвига данных выглядит наиболее реалистичной.

Такая ситуация в набольшей степени соответствует реальному использованию дискриминантной модели (классификатора) машинного обучения в критических применениях. Мы обучили модель на некотором тренировочном множестве (статический набор данных, слепок некоторого реального потока данных за некоторый интервал времени в прошлом), получили устраивающие нам метрики, подтвердили эти метрики на некотором тестовом наборе (наборах) данных, который также, естественно, есть исторический слепок реальных данных и запустили систему в реальную эксплуатацию. Теперь мы имеем реальный поток данных, каждый поступающий элемент которого и оценивается натренированным классификатором. В подавляющем большинстве случаев это будет M2M система, человек в контуре принятия решения отсутствует. Соответственно, истинных меток для реальных данных нет. Мы можем, в каких-то случаях, полагаться на решения, когда человек апостериори проверяет (оценивает) работу системы. Но это будет уже апостериорная оценка, выполняемая не в режиме реального времени. Что, в свою очередь, означает, что оценка сдвига данных должна выполняться автоматически. Результаты такой оценки должны формировать предупреждение о выходе модели из зоны проверенных данных, с возможностью для эксплуатантов прекратить (или продолжить) эксплуатацию модели.

В работе [11] как раз и были предложений именно непараметрические подходы. В работе предложены методы для обнаружение изменений в потоках данных путем исключения любых предположений, связанных с функцией плотности вероятности, которая генерирует эти измеренные значения. Вместо этого мы предполагаем, что поступающие реальные значения соответствуют независимым случайным переменным из ограниченного диапазона. Кроме того, предлагаемые методы приспособлены для онлайн-обучения (модель обучается по мере поступления новых данных): используют один проход для обработки, время обработки и используемая память - постоянны, гарантирована производительность в части ложных и ложноотрицательных срабатываний. Но при этом предполагается, что все входные данные помечены. Идея мониторинга заключается в использовании неравенства Хёфдинга (Hoeffding's inequality), которое даёт верхнюю границу вероятности того, что сумма случайных величин отклоняется от своего математического ожидания:

Пусть $X_1, X_2, \ldots X_n$ есть независимые переменные, такие что $X_i \in [a_i, b_i]$ где $i \in \{1,2,\ldots n\}$.

Пусть $\overline{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$ Тогда для любого $\varepsilon > 0$

$$\Pr\{\overline{X} - E[\overline{X}] \geq \varepsilon\} \leq e^{-2n^2\varepsilon^2 / \sum_{i=1}^{n}(b_i - a_i)^2}$$

Есть модели, в которых допускается частичное присутствие меток поступающих данных [12]. Метки могут запрашиваться для групп данных, внутри некоторого окна, либо при каких-то изменениях данных.

Работа [13] посвящена определению сдвига данных именно в неразмеченных потоках. Как отмечалось выше, сдвиг концепций можно разделить на две группы в зависимости от типа изменения: реальные и виртуальные. В реальном сдвиге, также называемом дрейфом класса или сдвигом априорной вероятности, изменение влияет на условные вероятности класса $P(Y \mid X)$, в то время как функции $P(X)$ могут меняться или нет. В виртуальном дрейфе, также называемое изменением признаков или ковариантным сдвигом, распределение признаков $P(X)$ меняется со временем, а границы классов остаются неизменными. Так, для классификаторов, виртуальные сдвиги не влияют на

результаты работы классификатора. Очевидно также, что для реальных дрейфов, необходимо, чтобы изменения также были видны во входных характеристиках *P(X)*. Когда изменения касаются условной вероятности *P(Y|X)* их невозможно определить без разметки реальных данных.

Детекторы дрейфа без использования информации о метках могут выглядеть так. У нас есть два набора данных (два окна) – один с историческими данными и второй – с текущими. Изначально историческое окно заполнено данными тренировочного набора. Далее проверяется нулевая гипотеза о том, что обе группы происходят из одного и того же распределения. Если гипотеза отвергается, то обнаружен дрейф. Далее исторические данные заменяются проверенным набором данных, а в качестве нового проверяемого набора выступают новые поступившие данные. Технически – это прозрачный и работающий механизм, но выше мы указывали на то, что это очень затратный по времени механизм. В [13] предлагается представлять (кодировать) значения признаков как оттенки серого. В итоге, для окна данных из *M* значений, каждое из которых имеет *N* атрибутов (features), мы получаем картинку размером *MxN*. Соответственно, такие два изображения (для исторических данных и для реальных) можно сравнить визуально.

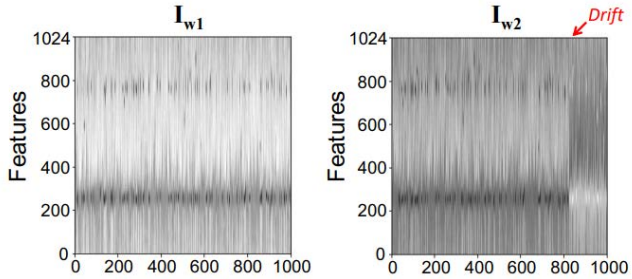

Рис. 5. Визуальное представление сдвига данных [13]

Но помимо визуального сравнения, можно сравнивать такие изображения алгоритмически, сравнивая интенсивности пикселей. Метрика для сравнения – среднеквадратичное отклонение (Mean-Squared Deviation или MSD). Для изображений размером *m x n* эта метрика вычисляется так

$$MSD\ (I_1, I_2) = \frac{1}{m \times n} \sum_{i=1}^{n} \sum_{j=1}^{m} (I_1(i,j) - I_2(i,j))^2$$

Общая идея MSD заключается в измерении разницы между интенсивностью пикселей двух изображений. MSD обладает свойством неотрицательности, а значение 0 указывает на полное сходство между сравниваемыми изображениями. MSD симметрично, а это означает, что $MSD(I_1, I_2) = MSD(I_2, I_1)$.

В соответствии со значением сходства, возвращаемым MSD, мы можем отметить дрейф, когда последовательность со значениями *k* выше верхнего порога или ниже нижнего порога. Причем эти пределы могут динамически меняться во время работы приложения.

IV. Как практически выглядит мониторинг сдвига данных?

Например, при использовании sckit-multiflow это выглядит примерно так:

```
import numpy as np
from
skmultiflow.drift_detection.hddm_w
import HDDM_W
# method for stream detection
hddm_w = HDDM_W()
# Simulating a data stream as a normal
distribution of 1's and 0's
data_stream = np.random.randint(2,
size=2000)
# Changing the data concept from index
999 to 1500, simulating an
# increase in error rate
for i in range(999, 1500):
    data_stream[i] = 0
# Adding stream elements to HDDM_A and
verifying if drift occurred
for i in range(2000):
    hddm_w.add_element(data_stream[i])
    if hddm_w.detected_warning_zone():
     print('Warning zone has been
     detected in data: ' +
     str(data_stream[i]) + ' - of
     index: ' + str(i))
    if hddm_w.detected_change():
     print('Change has been detected
     in data: ' + str(data_stream[i])
     + ' - of index: ' + str(i))
```

Другие методы, которые поддерживаются в этом пакете:

- Adaptive Windowing – анализ сдвига концепций на базе адаптивного окна.
- DDM - Drift Detection Method.
- EDDM- модификация DDM.
- HDDM_A – анализ сдвига на основе метода Хёфдинга с скользящей средней.
- HDDM_W - анализ сдвига на основе метода Хёфдинга с взвешенной скользящей средней.
- KSWIN - Kolmogorov-Smirnov Windowing критерий Колмогорова-Смирнова.
- PageHinkley - Page-Hinkley method for concept drift detection

Для специализированных пакетов мониторинга это может выглядеть еще проще. Например, Evidently AI [14]. Это один из инструментов с открытым исходным кодом, который может использоваться для мониторинга работы моделей машинного обучения.

```
data_drift_report = Report(metrics=[
    DataDriftPreset(),
])
data_drift_report.run(current,
reference_data, column_mapping=None)
```

То есть, просто сравниваются два набора данных (два окна на реальном наборе данных и тренировочном

наборе данных)

Отметим также, что инструментарий включает в себя средства визуализации сдвига данных по объясняющим переменным. Это выглядит примерно так:

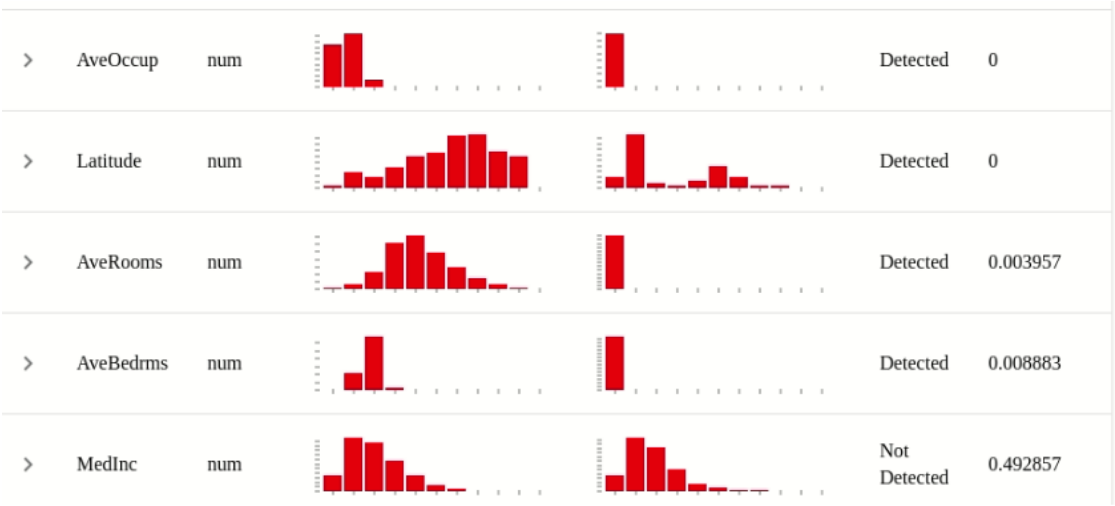

Рис. 6. Визуализация в Evidently AI

Fiddler AI Monitoring [15] имеет набор инструментов, которые помогают в объяснении решений модели и позволяют вести мониторинг моделей на этапе эксплуатации. Обнаружение сдвига данных – одна из поддерживаемых опций.

Fiddler использует JS-дивергенцию и объяснимый искусственный интеллект, чтобы помочь выявить проблемы модели и возможный сдвиг данных за несколько шагов:

Шаг 1. Определяем дрейф прогноза в выходных данных модели в реальном времени по сравнению с обучающим или базовым набором; рассчитанную JS-дивергенцию соотносим со знаниями предметной области о пороговых значениях дрейфа в реальных данных.

Шаг 2. В выбранном временном окне проверяем дрейф основных характеристик, снова используя JS-дивергенцию. Используем объяснимость для просмотра важности дрейфующих характеристик и отбираем те, которые имеют наиболее значимый эффект.

Шаг 3. Сравниваем распределение данных, чтобы увидеть фактическую разницу и сформировать решение о необходимости переобучения модели.

Отметим, что здесь уже фигурируют знания о предметной области. Это, естественно, не формализуемо в общем случае, но понятно, например, что измерения могут иметь некоторые естественные границы измерения.

Встроенная визуализация также является полезным (и обязательным для такого рода инструментов) механизмом.

Майкрософт предоставляет автоматизированный способ определения смещения данных, интегрированный во фреймворк Azure ML [16]. Azure ML использует статистические методы для определения смещения, при этом для расчета смещения для выбранных функций используются разные временные окна.

Есть один важный момент во всех реализациях. Для сравнения (определения сдвига данных) необходим образец для сравнения. Что естественно. Но как это выглядит, если нам необходимо исполнять модель на некоторой встроенной вычислительной системе? Это опять-таки типичная картина для критических приложений. Модель тренируется и исполняется (inference) на разных системах. Но это означает, что на встроенной платформе, помимо самой сериализованной модели для исполнения, нужно будет разместить и сравнительный датасет, использованный для обучения. Теоретически (и практически) встроенная система может быть утрачена. И датасет, оказавшийся в чужих руках, может быть использован для тренировки теневой модели и создания состязательных атак. Этот вопрос нигде не поднимался, и задачи, например, шифрования сравнительного набора, не рассматриваются.

Рис. 7. Визуализация в Fiddler AI

V О ПРОГРАММНОЙ АРХИТЕКТУРЕ СИСТЕМЫ МОНИТОРИНГА

Архитектуру системы мониторинга предлагается организовать по аналогии с Ламбда-моделью в обработке данных [17]. Рабочая версия модели получает и обрабатывает данные в реальном времени. Параллельно, те же данные получает цифровой двойник модели. Двойник для проверки сдвига данных и возможного дообучения модели работает уже не в реальном времени, отсюда появляется очередь перед его входом. И рабочая модель (веса нейронной сети) в произвольный момент времени может быть обновлена двойником.

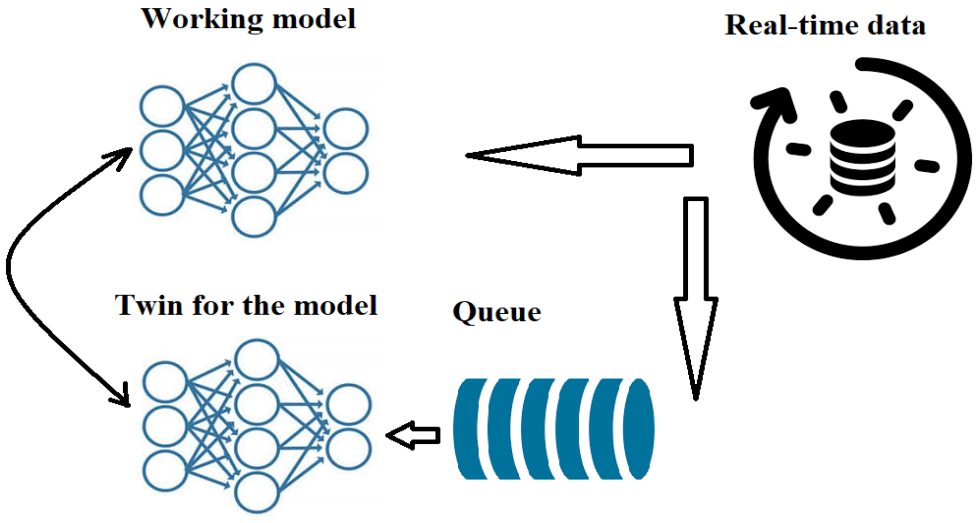

Рис. 8. Архитектура системы мониторинга

Благодарности

Мы благодарны сотрудникам кафедры Информационной безопасности факультета ВМК МГУ имени М.В. Ломоносова за ценные обсуждения. Работа является частью проекта по созданию доверенной платформы машинного обучения [18].

Исследование выполнено при поддержке Междисциплинарной научно-образовательной школы Московского университета «Мозг, когнитивные системы, искусственный интеллект».

Библиография

[1] Dong, Guozhu, and Huan Liu, eds. Feature engineering for machine learning and data analytics. CRC Press, 2018.
[2] Ilyushin, Eugene, Dmitry Namiot, and Ivan Chizhov. "Attacks on machine learning systems-common problems and methods." International Journal of Open Information Technologies 10.3 (2022): 17-22.
[3] Namiot, Dmitry, and Eugene Ilyushin. "On the robustness and security of Artificial Intelligence systems." International Journal of Open Information Technologies 10.9 (2022): 126-134.
[4] Kupriyanovsky, V., and D. Namit. "Digital economy-Smart way to work." International Journal of Open Information Technologies 2.4 (2016): 26-32.
[5] Namiot, Dmitry, Eugene Ilyushin, and Ivan Chizhov. "The rationale for working on robust machine learning." International Journal of Open Information Technologies 9.11 (2021): 68-74.
[6] Understanding Dataset Shift and Potential Remedies https://vectorinstitute.ai/wp-content/uploads/2021/08/ds_project_report_final_august9.pdf
[7] Gama, João, et al. "A survey on concept drift adaptation." ACM computing surveys (CSUR) 46.4 (2014): 1-37.
[8] Baena-Garcıa, Manuel, et al. "Early drift detection method." Fourth international workshop on knowledge discovery from data streams. Vol. 6. 2006.
[9] Zheng, Shihao, et al. "Labelless concept drift detection and explanation." NeurIPS 2019 Workshop on Robust AI in Financial Services: Data, Fairness, Explainability, Trustworthiness, and Privacy. 2019.
[10] Ma, Sisi, and Roshan Tourani. "Predictive and causal implications of using shapley value for model interpretation." Proceedings of the 2020 KDD Workshop on Causal Discovery. PMLR, 2020.
[11] Frias-Blanco, Isvani, et al. "Online and non-parametric drift detection methods based on Hoeffding's bounds." IEEE Transactions on Knowledge and Data Engineering 27.3 (2014): 810-823.
[12] Žliobaitė, Indrė, et al. "Active learning with drifting streaming data." IEEE transactions on neural networks and learning systems 25.1 (2013): 27-39.
[13] Souza, Vinicius MA, Farhan A. Chowdhury, and Abdullah Mueen. "Unsupervised drift detection on high-speed data streams." 2020 IEEE International Conference on Big Data (Big Data). IEEE, 2020.
[14] Evidently AI https://www.evidentlyai.com/
[15] Fiddler AI https://www.fiddler.ai/blog/how-to-detect-data-drift
[16] Kenthapadi, Krishnaram, et al. "Model Monitoring in Practice: Lessons Learned and Open Challenges." Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022.
[17] Namiot, Dmitry, Manfred Sneps-Sneppe, and Romass Pauliks. "On data stream processing in IoT applications." Internet of Things, Smart Spaces, and Next Generation Networks and Systems. Springer, Cham, 2018. 41-51.
[18] Namiot, Dmitry, Eugene Ilyushin, and Oleg Pilipenko. "On Trusted AI Platforms." International Journal of Open Information Technologies 10.7 (2022): 119-127.

Data shift monitoring in machine learning models

Dmitry Namiot, Eugene Ilyushin

***Abstract* — The fundamental moment of the operation of machine learning systems is that the models are trained on some selected training data set. Accordingly, the generalizations obtained at the training stage are due to the characteristics of some subset of the general population. If the characteristics of the data change during the operation of the system, then generalizations of the model become, generally speaking, untenable. At the same time, such a change in data should be considered the rule rather than the exception. This change in data characteristics is called data shift. This, in turn, means that any machine learning system that claims to be industrial must track the possible data shift. The presence of such a shift reduces the confidence in the results of the work or even makes the system unsuitable for further operation. Taking into account (overcoming) such a data shift is a separate task, simple retraining can be a big problem for critical applications, for example. But in any case, the first task is to determine the fact of data shift. The data shift itself is divided into several types, the most serious of which is a change in the relationship between dependent and independent variables. Naturally, the definition of data offset for streams is of particular interest, since this is directly related to critical applications.**

***Keywords* – machine learning, monitor, data shift**

REFERENCES

[1] Dong, Guozhu, and Huan Liu, eds. Feature engineering for machine learning and data analytics. CRC Press, 2018.

[2] Ilyushin, Eugene, Dmitry Namiot, and Ivan Chizhov. "Attacks on machine learning systems-common problems and methods." International Journal of Open Information Technologies 10.3 (2022): 17-22.

[3] Namiot, Dmitry, and Eugene Ilyushin. "On the robustness and security of Artificial Intelligence systems." International Journal of Open Information Technologies 10.9 (2022): 126-134.

[4] Kupriyanovsky, V., and D. Namit. "Digital economy-Smart way to work." International Journal of Open Information Technologies 2.4 (2016): 26-32.

[5] Namiot, Dmitry, Eugene Ilyushin, and Ivan Chizhov. "The rationale for working on robust machine learning." International Journal of Open Information Technologies 9.11 (2021): 68-74.

[6] Understanding Dataset Shift and Potential Remedies https://vectorinstitute.ai/wp-content/uploads/2021/08/ds_project_report_final_august9.pdf

[7] Gama, João, et al. "A survey on concept drift adaptation." ACM computing surveys (CSUR) 46.4 (2014): 1-37.

[8] Baena-Garcıa, Manuel, et al. "Early drift detection method." Fourth international workshop on knowledge discovery from data streams. Vol. 6. 2006.

[9] Zheng, Shihao, et al. "Labelless concept drift detection and explanation." NeurIPS 2019 Workshop on Robust AI in Financial Services: Data, Fairness, Explainability, Trustworthiness, and Privacy. 2019.

[10] Ma, Sisi, and Roshan Tourani. "Predictive and causal implications of using shapley value for model interpretation." Proceedings of the 2020 KDD Workshop on Causal Discovery. PMLR, 2020.

[11] Frias-Blanco, Isvani, et al. "Online and non-parametric drift detection methods based on Hoeffding's bounds." IEEE Transactions on Knowledge and Data Engineering 27.3 (2014): 810-823.

[12] Žliobaitė, Indrė, et al. "Active learning with drifting streaming data." IEEE transactions on neural networks and learning systems 25.1 (2013): 27-39.

[13] Souza, Vinicius MA, Farhan A. Chowdhury, and Abdullah Mueen. "Unsupervised drift detection on high-speed data streams." 2020 IEEE International Conference on Big Data (Big Data). IEEE, 2020.

[14] Evidently AI https://www.evidentlyai.com/

[15] Fiddler AI https://www.fiddler.ai/blog/how-to-detect-data-drift

[16] Kenthapadi, Krishnaram, et al. "Model Monitoring in Practice: Lessons Learned and Open Challenges." Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. 2022.

[17] Namiot, Dmitry, Manfred Sneps-Sneppe, and Romass Pauliks. "On data stream processing in IoT applications." Internet of Things, Smart Spaces, and Next Generation Networks and Systems. Springer, Cham, 2018. 41-51.

[18] Namiot, Dmitry, Eugene Ilyushin, and Oleg Pilipenko. "On Trusted AI Platforms." International Journal of Open Information Technologies 10.7 (2022): 119-127.