**Проект: Анализ пользовательского поведения на сайте** СберАвтоподписка для прогнозирования целевых действий

Команда ТЕАМ-А:

* Бортяков Алексей Юрьевич
* Горозий Максим Юрьевич
* Козырев Дмитрий Анатольевич
* Решетило Никита Сергеевич

Задача компании «СберАвтоподписка»: Повышение эффективности сайта через улучшение пользовательского опыта, рост конверсии и результативности рекламных кампаний.

Наша задача: Создать модель машинного обучения, предсказывающую вероятность совершения пользователем целевого действия на сайте.

Ожидаемый результат и польза для компании: Реализованная модель позволит:

* Оценивать эффективность различных каналов привлечения трафика.
* Адаптировать и оптимизировать рекламные кампании.
* Улучшать пользовательский опыт (UX) на сайте на основе анализа поведения посетителей.

Основные этапы работы над проектом:

1. Загрузка и первичное исследование данных (EDA).
2. Предобработка данных: Очистка, агрегация и создание новых признаков.
3. Выбор, обучение и оценка модели.
4. Разработка API и деплоймент решения.

Обзор Данных и Первичное Исследование (EDA)

Мы работали с двумя основными наборами данных, предоставленными компанией: ga\_sessions и ga\_hits.

Датасет ga\_sessions: Содержит информацию о пользовательских сессиях.

* Объем: 1,860,042 строк, 18 столбцов.
* Типы данных: В основном категориальные (object), 1 числовой (visit\_number).
* Ключевые наблюдения EDA:
  + Пропуски данных: Значительное количество пропусков в 'utm\_campaign' (~11.8%), 'utm\_adcontent' (~18.0%), 'device\_os' (~57.5%), 'device\_brand' (~19.7%). Поля 'device\_model' (~99.1%) и 'utm\_keyword' (~58.2%) содержат критически много пропусков, что ставит под вопрос их полезность.
  + Распределение визитов: Медианное количество визитов на пользователя — 1. Максимальное значение (564) — аномалия (возможно, бот или технический артефакт.
  + Идентификаторы: 'session\_id' уникален для каждой строки (1:1 с сессиями), 'client\_id' показывает, что есть повторные визиты пользователей (около 75% уникальных).
  + География и Устройства: Данные по стране ('geo\_country': ~97% Россия), категории устройства ('device\_category': ~79% mobile) и браузеру ('device\_browser': ~54% Chrome) заполнены хорошо и информативны.
  + Временные данные: 'visit\_date' и 'visit\_time' без пропусков, но выявлены аномалии в распределении (пики по датам, странное скопление в 12:00:00).

Датасет ga\_hits: Содержит детальную информацию о каждом действии (хите) пользователя внутри сессии.

* Объем: 15,726,470 строк, 11 столбцов. Значительно больше строк, чем в ga\_sessions, так как одна сессия включает много хитов.
* Типы данных: Числовые (hit\_time, hit\_number, event\_value), категориальные.
* Ключевые наблюдения EDA:
  + Пропуски данных: Критически много пропусков в 'event\_value' (100%) и 'hit\_time' (~58.25%). 'hit\_referer' (~39.9%) и 'event\_label' (~23.9%) также имеют значительные пропуски.
  + Тип хита: Поле 'hit\_type' содержит только одно уникальное значение ("event") и бесполезно для анализа.
  + Разнообразие событий: Поля 'hit\_page\_path' (уникальные URL), 'event\_category' и 'event\_action' содержат большое разнообразие значений, описывающих действия пользователя.
  + Связь с сессиями: Объединение с ga\_sessions возможно по 'session\_id'. В среднем, одна сессия содержит около 9 хитов.

Выводы по EDA: Данные содержат много пропусков и некоторые аномалии, требующие тщательной предобработки. Для прогнозирования целевого действия необходимо объединить данные из двух источников и извлечь информативные признаки о поведении пользователя в рамках всей сессии.

Предобработка Данных и Конструирование Признаков (Feature Engineering)

Основная цель этого этапа — трансформировать сырые данные из таблиц сессий и хитов в единый набор признаков, описывающих каждую пользовательскую сессию, для последующего использования моделью машинного обучения.

1. Определение Целевой Переменной: На основе анализа данных наиболее релевантным для прогнозирования целевого действия пользователя был выбран признак event\_action из датасета ga\_hits.
2. Обработка Пропусков и Очистка:
   * Все явные маркеры пропусков в данных (например, строковые значения "set", "none") были заменены на стандартное значение пропущенных данных NaN.
   * Для признаков 'hit\_referer' и 'event\_label' была применена специфическая логика заполнения пропусков:
     + Сначала для каждого пропуска внутри сессии предпринималась попытка найти первое непустое значение этого признака среди всех хитов на той же странице (hit\_page\_path) или с тем же действием (event\_action).
     + Если такое значение не находилось, оставшиеся пропуски заполнялись константными значениями: 'hit\_referer' — 'direct' (прямой заход), 'event\_label' — 'none' (отсутствие метки). Это позволило максимально использовать доступную контекстную информацию внутри сессии.
3. Кодирование Категориальных Признаков:
   * Для работы с категориальными данными, включая те, где были пропуски, использовалась функция кодирования.
   * Эта функция преобразует уникальные строковые значения в числовые коды. Пропущенным значениям (NaN) присваивается код 0, а всем уникальным категориям — последовательные целые числа, начиная с 1.
4. Агрегация Данных (Feature Engineering): Этот этап стал ключевым для преобразования детализированных данных о хитах в признаки уровня сессии. Данные из ga\_hits были сгруппированы по session\_id, и к различным признакам применены агрегирующие функции. Результат этой агрегации был затем объединен с данными из ga\_sessions.

Примеры агрегированных признаков:

|  |  |  |  |
| --- | --- | --- | --- |
| Исходный признак (df.columns) | Примененная агрегация (Правило в agg\_rules) | Итоговый признак (название после переименования) | Что описывает признак в рамках сессии |
| hit\_number | min, max, count | first\_hit\_number, last\_hit\_number, total\_hits | Порядковый номер первого и последнего хита, общее количество хитов в сессии. |
| hit\_page\_path | Первое значение (iloc[0]) | entry\_page | Входная страница сессии (страница первого хита). |
| event\_category\_grouped | Наиболее частое значение (value\_counts().index[0]) | main\_category\_grouped | Основная (наиболее часто встречающаяся) категория событий в сессии. |
| event\_action\_grouped | Наиболее частое значение (value\_counts().index[0]) | main\_action\_grouped | Основное (наиболее часто встречающееся) действие событий в сессии. |
| hit\_time\_2 | Сумма (sum) | total\_time | Суммарное значение признака hit\_time\_2 за всю сессию (его точный смысл зависит от того, что означает hit\_time\_2). |
| hit\_referer\_encoded | Наиболее частое значение (value\_counts().index[0]) | main\_referer | Основный (наиболее часто встречающийся) источник перехода (реферер) в сессии (в закодированном виде). |
| event\_label\_encoded | Наиболее частое значение (value\_counts().index[0]) | main\_label | Основная (наиболее часто встречающаяся) метка события в сессии (в закодированном виде). |
| Признаки, начинающиеся на utm\_ | Первое значение (iloc[0]) | Исходное имя признака (utm\_source, utm\_medium и т.д.) | Значение UTM-метки из первого хита сессии. |

Также были созданы временные признаки из даты сессии (month, day, day\_of\_week\_num).

1. Финальный Датасет: После объединения агрегированных данных хитов с данными сессий и удаления неинформативных признаков, а также оптимизации типов данных, был получен финальный датасет для обучения модели.
   * Количество сессий: 1860042
   * Количество признаков: 22.
   * Используемая память: Оптимизировано до ~115 MB.

Структура финального датасета включает признаки, описывающие характеристики сессии (номер визита, UTM-метки, устройство, гео) и агрегированные поведенческие метрики (количество хитов, длительность, входная страница, основные типы событий/действий, реферер).

Выбор и Обучение Модели

Выбор Модели: Для решения задачи многоклассовой классификации (прогнозирования различных типов event\_action) мы выбрали LightGBM (Light Gradient Boosting Machine).

Обоснование выбора LightGBM:

* Скорость и Эффективность: LightGBM известен своей высокой скоростью обучения и низким потреблением памяти по сравнению с другими реализациями градиентного бустинга (как XGBoost), особенно на больших табличных данных. Он использует эффективные алгоритмы построения деревьев (Gradient-based One-Side Sampling и Exclusive Feature Bundling).
* Производительность: Модели на основе бустинга деревьев часто показывают превосходные результаты на структурированных данных, и LightGBM является одной из ведущих библиотек в этой области.
* Масштабируемость: Хорошо подходит для работы с датасетами размера, как у нас.

Как работает LightGBM (кратко):

LightGBM строит ансамбль решающих деревьев последовательно. Каждое новое дерево обучается на ошибках (градиентах) предсказаний, сделанных всеми предыдущими деревьями. Итоговое предсказание получается путем взвешенной суммы предсказаний всех деревьев.

Конфигурация и Гиперпараметры Модели: Гиперпараметры LightGBM были подобраны экспериментально для достижения наилучшего баланса между качеством модели и скоростью обучения.

* random\_state (42): Фиксирует генератор случайных чисел для обеспечения воспроизводимости результатов обучения.
* n\_estimators (150): Количество решающих деревьев в ансамбле. Определяет общее количество итераций бустинга. Увеличение может повысить точность, но потенциально ведет к переобучению и увеличивает время обучения.
* learning\_rate (0.2): Скорость обучения. Определяет вклад каждого нового дерева в итоговое предсказание. Меньшие значения требуют больше n\_estimators, но могут улучшить качество; большие значения ускоряют обучение, но менее стабильны.
* max\_depth (9): Максимальная глубина каждого дерева. Ограничивает сложность отдельного дерева. Большая глубина позволяет модели улавливать более сложные взаимодействия, но увеличивает риск переобучения.
* num\_leaves (31): Максимальное количество листьев (конечных узлов) в дереве. Это ключевой параметр в LightGBM, контролирующий сложность дерева. Большее число листьев делает модель более гибкой. Значение по умолчанию часто является хорошей отправной точкой.
* subsample (0.8): Доля данных (строк), случайным образом отбираемая для обучения каждого дерева. Используется для борьбы с переобучением (метод "бэггинга"). 80% данных используются для каждого дерева.
* colsample\_bytree (0.8): Доля признаков (столбцов), случайным образом отбираемая для обучения каждого дерева. Еще один метод для предотвращения переобучения и ускорения, фокусирует каждое дерево на подмножестве признаков. 80% признаков используются для каждого дерева.
* reg\_alpha (0.1, L1): Параметр L1-регуляризации. Добавляет штраф за абсолютные значения весов, способствуя разреженности модели (обнуление весов для менее важных признаков).
* reg\_lambda (0.1, L2): Параметр L2-регуляризации. Добавляет штраф за квадраты весов, помогая предотвратить переобучение, делая веса менее экстремальными.
* n\_jobs (-1): Количество потоков/ядер процессора, используемых для обучения. Значение -1 задействует все доступные ядра для ускорения.
* verbose (-1): Уровень вывода диагностической информации в процессе обучения. -1 отключает вывод.
* boosting\_type ('gbdt'): Алгоритм бустинга. 'gbdt' (Gradient Boosting Decision Tree) является стандартным алгоритмом.

Результаты Обучения:

* Обучено на 18 классах целевой переменной (event\_action).
* Метрика оценки: ROC-AUC (подходит для многоклассовой классификации и оценки качества предсказания вероятностей).
* ROC-AUC на тренировочных данных: 0.9476
* ROC-AUC на тестовых данных: 0.9251 Полученные метрики свидетельствуют о хорошем качестве модели и ее способности обобщать на невидимых данных, предсказывая вероятность различных целевых действий.

Результаты Анализа, Реализация и Выводы

Ключевые Инсайты из Обработанных Данных: Анализ финального датасета, обогащенного агрегированными поведенческими признаками, подтвердил и углубил первичные выводы:

* Доминирующие входные страницы: Около 50% сессий начинаются с главной страницы подписки (podpiska.sberauto.com/), что указывает на основной пользовательский сценарий.
* Поведенческие паттерны: Наиболее частые действия связаны с просмотром карточек автомобилей (около 55% от основных действий) и использованием поисковых форм (9%). Это ключевые этапы пользовательского пути.
* Распределение трафика: Подтверждена высокая концентрация трафика из России (Москва + СПб составляют почти 60%). Основные каналы привлечения - баннерная реклама, контекстная реклама (CPC) и органический поиск.
* Глубина сессии: В среднем сессия содержит около 9 хитов, что дает представление об уровне вовлеченности пользователей.

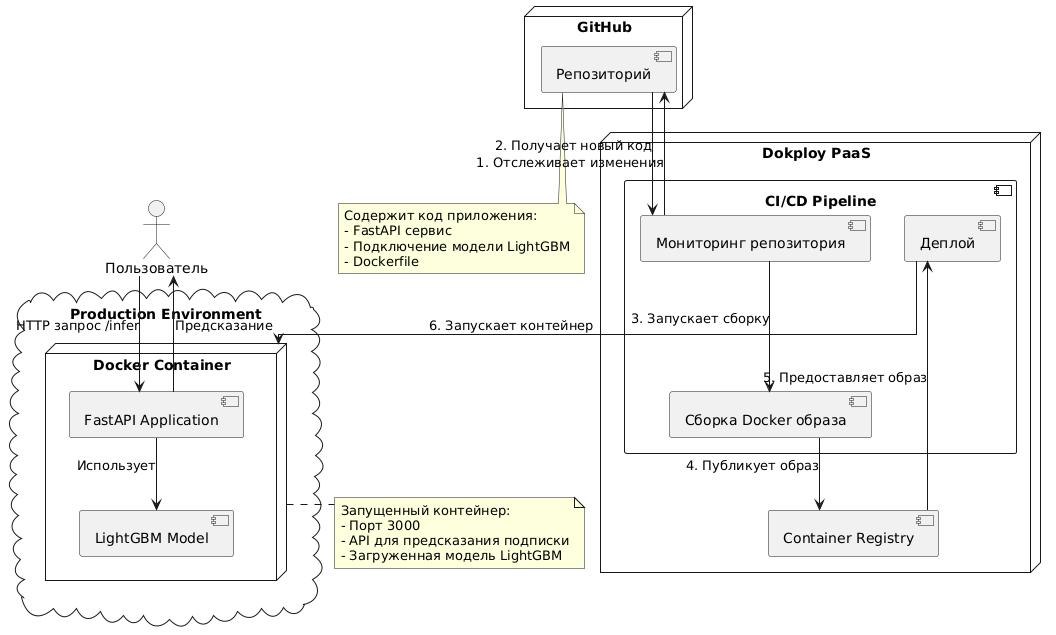
Эти инсайты могут быть непосредственно использованы для оптимизации UX, таргетирования рекламы и оценки эффективности каналов.

Реализация Решения: Для интеграции модели в существующую или будущую инфраструктуру компании было разработано готовое решение:

1. Экспорт Модели: Обученная модель LightGBM была сохранена в формате pickle для удобства использования.
2. Разработка Веб-сервиса (API): Создан микросервис на фреймворке FastAPI. Этот сервис:
   * Принимает входные данные для одной или нескольких сессий.
   * Валидирует входные данные на соответствие ожидаемому формату.
   * Использует загруженную модель для выполнения предсказаний (.predict\_proba()).
   * Возвращает предсказанные вероятности целевых действий.
3. API Документация: Для облегчения тестирования и интеграции сервиса с другими системами была автоматически сгенерирована документация в формате OpenAPI 3 (с доступным интерфейсом Swagger UI).

**Развертывание (Deployment):**

* При разработке и развертывании прототипа мы сосредоточились на быстром получении production-ready решения.
* Использование стандартного стека *nginx + gunicorn* было отвергнуто, так как реализация сопутствующей инфраструктуры заняла бы слишком много времени.
* Решением стало использование PaaS (Platform as a Service). На российском рынке подобных решений в настоящий момент нет.
* Был опробован PaaS-сервис Vercel, но ML-приложение не смогло работать в рамках ограничений бесплатного тарифа.
* Чтобы сохранить удобство и автоматизацию процессов развертывания, логирования, резервного копирования, версионирования и т. д., характерные для PaaS-платформ, на VPS была установлена и настроена open-source PaaS-система Dokploy.



Выводы: В рамках хакатона наша команда успешно прошла все этапы машинного обучения: от загрузки и глубокого анализа данных, выявления проблем и аномалий, до комплексной предобработки и конструирования информативных признаков, выбора, обучения и оценки производительной модели LightGBM. Созданное решение представлено в виде готового к интеграции веб-сервиса с API, развернутого на собственной инфраструктуре с использованием Dokploy для эффективного управления. Модель демонстрирует высокие показатели качества предсказания вероятности целевых действий, что делает ее ценным инструментом для повышения эффективности сайта СберАвтоподписка.

Дальнейшие шаги (опционально):

* Сбор дополнительных данных для обогащения признаков.
* Исследование других моделей или ансамблевых подходов.
* A/B тестирование модели на реальных пользователях.