Описание программы:

После чтения данных из таблиц transactions и nomenclature и соединения их в одну таблицу (df) по столбцу sku_id, я вывожу график популярности категорий, и располагаю нужные для рекомендаций категории в списке (по популярности). Затем для каждой категории я запоминаю 20 самых популярных товаров. В таблице df оставляю только два столбца – sku_id и cheque_id. Почему только два? Во-первых – для уменьшения нагрузки и сокращения работы программы, а во-вторых – добавление некоторых категорий (страна, признак собств. торговой марки) не меняло коэффициент точности угадывания на тестовых данных, конечно стоило проверить комбинации признаков и каждый признак по отдельности и его влияние на точность, но к сожалению, мне не хватило времени и ресурсов.

Далее я выделяю категории и добавляю для каждой отдельный столбец со значениями [0,1], sku_id удаляется. Затем группирую данные по айди чека и складываю значения всех столбцов (чтобы одна строка показывала все содержимое чека).

Читаю и привожу в такой же вид данные transactions-for_submission. (Т.к мой алгоритм получился слишком трудоемкий, то я обрезаю таблицу данных для рекомендаций ☹. 1000 строк алгоритм проходит за ~28секунд, и следовательно для обработки всей таблицы нужно ~5 часов).

После этого для каждой категории я строю свое дерево решений. Глубина дерева (1, 2, 3) – также не показала каких-либо значительных различий на тестовых данных, поэтому я оставила глубину 1.

Далее для каждого чека из transactions-for_submission мы предсказываем значения для наших категорий и если предсказание != 0, то запоминаем эту категорию для данного чека. После предсказаний обходим категории, которые мы запомнили, и добавляем в список рекомендаций примерно равное количество товаров из каждой (т.к. категории и товары выстроены по популярности – то и в списке рекомендаций они будут выстроены по этому признаку).

Идея, которую я не успеваю реализовать: для каждого клиента с картой покупателя, анализировать его предыдущие покупки и делать вывод на основании этого. От самого простого – взять массив товаров, которые он уже приобретал, вычесть из него текущие товары из чека и порекомендовать самые популярные из оставшихся, до обучения модели и прогноза для каждого постоянного пользователя отдельно (в случае, когда данных недостаточно, возвращаться к предыдущему методу или комбинировать их).

Возможно, хорошей идеей было бы прогнозировать покупку самих товаров, а не выбор категории и рекомендацию самого популярного товара в ней. Все-таки, когда мы рекомендуем самый популярный товар, не учитываются желания и потребности каждого отдельного клиента, ведь непопулярные товары тоже покупаются.

Это мой первый опыт работы с машинным обучением и большими данными, поэтому если бы времени у меня было больше, то я конечно бы глубже изучила данную тему и проанализировала разные модели для рекомендательной системы. Из-за того, что я прохожу отбор еще на несколько направлений стажировки, мне удалось углубленно заниматься заданием только 4 дня. Это было довольно сложно, но очень интересно и познавательно. Спасибо, что дали возможность попробовать себя в новой для меня сфере!

*В папке data лежат все файлы .parquet, в том числе файл-решение с рекомендациями (для первых 50000 строк от изначальной таблицы transactions-for_submission). Файл с кодом программы лежит в файле “task_gasprom.ipynb”, и надеюсь, что я выбрала правильный формат для сохранения проекта Jupiter Notebook и у Вас не возникнет проблем при просмотре.