Техническое задание:

1.Введение:

Заголовок задачи	Разработка интерпретируемой скоринговой модели для оценки кредитоспособности клиентов сегмента PDL
Контекст	*В рамках кредитного конвейера необходимо создать модуль принятия решений по клиентам о выдаче кредита. В качестве модуля принятий решений выступает скрипт разработанной скоринговой модели.*
Проблема / постановка задачи	*Необходимо автоматизировать процесс принятия решений по выдаче кредитов, опираясь на исторические данные по клиентам.* *Цель задачи: создать механизм ранжирования клиентов по вероятности выхода в дефолт.* *В качестве обязательного базового решения задачи необходимо сгенерировать и описать банк переменных на полученных сырых данных, отобрать топ переменных и построить модель логит регрессии – классическую скоринговую карту, учитывая ограничения из Технического задания.* *Дополнительно, помимо базового решения, приветствуются модели-кандидаты с использованием любых продвинутых методов (леса, бустинги, нейросети).*
Требования к используемому ПО и к инструментам и методам реализации	• *Python (среда в Anaconda/Miniconda);* • *Обязательные библиотеки: Pandas, Numpy, Scikit-learn, OS;* • *Дополнительные библиотеки: LightAutoML, Catboost/XGBoost/LightGBM, OptBinning , Mlxtend, Shap;* • *Базовые знания Linux команд и Bash;* *Базовые знания Docker будут плюсом.*
Необходимая и доступная для решения информация	• *Обезличенные датасеты с кредитной историей, описанием полей и таргетом для моделирования;* • *Краткое техническое задание по задаче с ограничениями и запретами;* • *Метрики необходимого качества моделей и переменных.* •
Образ результата	*В каком виде вы хотите забрать результаты:* • *Прототип и скрипты;* • *Документация;* *Тестовые выборки, отскоренные полученной моделью.*

2. Описание файлов

№	Файл	Назначение
1	mapping.xlsx	Описание сырых данных
2	df_BKI_30k.csv	- Анонимизированный датасет с сырыми данными по кредитным историям (КИ) для создания модельных переменных. - В датасете представлены все кредиты клиентов, отобранных для кейса. - Формат датасета: одна строка – один кредит, подтянутый к заявке. - Кодировка utf-8
3	df_target_30k.csv	- Анонимизированный датасет с таргетом (таргет – выход в просрочку: target = 0 –хороший клиент, target = 1 – плохой клиент) Формат датасета: одна строка – одна заявка. Ключ для join application_id, client_id - Кодировка utf-8
4	df_test_notarget_10k.csv	- Анонимизированный датасет с сырыми данными по кредитным историям (КИ) для скоринга итоговой моделью для теста заказчиком. - Кодировка utf-8

3. Основные этапы моделирования по базовой задаче:

1) Анализ входных данных

На данном этапе необходимо:

- Ознакомиться с описанием сырых данных из файла mapping;
- Посчитать статистики по миссингам, аномалиям в данных;
- В случае, если вы решите произвести очистку или имплементацию миссингов / аномалий, исключений столбцов данных из анализа и т.д., то необходимо все обосновать и задокументировать подробно процесс.

2) Создание датасетов:

- Генерация банка сырых переменных – фичей на основе сырых данных
- Далее агрегация этих фичей в формате одна заявка – одна строка (агрегация по заявке);
- Составить физическое описание фичей хотя бы на уровне общих категорий (фичи, относящиеся к дням просрочки/просроченным суммам/кол-ву кредитов/платежей и тд), соблюдать чистоту кода, оставлять комментарии;
- Соблюдать физический смысл при создании фичей. Избегать бессмысленного перемножения фич. Например, умножения количества залогов на сумму кредита. Каждая фича должна иметь смысл, под который можно подогнать логику кредитного анализа. Например, динамика просроченных платежей по кредиту, выплаченная сумма к телу кредита и тд.;
- Расчет фичей производится на df_target_30k.csv, на df_test_notarget_10k.csv только накатываете скрипт фичей.

3) Объединение датасетов с фичами и таргетами:

- После агрегации датасэмплов с данными сджойнить по ключу таргеты (преобразованный df_target_30k.csv джойните с df_target_30k.csv).

4) Разбивка датасета на трейн/тест:

- агрегированный 30-ти тысячный сэмпл (из шага 3) разбить на свои

трейн и тест (пропорции вариативны, обычно это 70/30). Таким образом у вас будет размеченный тест;
- Не забудьте зафиксировать random_seed при сплите!

5) **Биннинг и группировка (WOE-преобразование) фичей:**
- WOE биннинг рассчитываете **только на трейн-выборке**, **на полученной размеченной тест-выборке просто накатываете;**
- Рекомендую библиотеку Optbinning (https://gnpalencia.org/optbinning/);
- Ограничить минимальный размер бина на трейне 5-ю процентами популяции;
- Кол-во бинов в одной переменной не более 5-6;
- Соблюдать риск логику: там, где бэдрейт по логике должен быть монотонным, он должен оставаться монотонным (например, чем больше процент просроченных платежей, тем выше бэдрейт в группе). Там, где это правило нарушено, нужно обосновать (например, возраст клиента имеет U-shape по бэдрейту);
- Файлы (joblib/pickle) с биннингом или, если у вас это будет скрипт, обязательно сохраняем и прикрепляем к кейсу;
- Накатываем полученный woe биннинг на все три выборки (трейн/тест и неразмеченная выборка).

6) **Однофакторный анализ (IV, Gini, PSI):**
Далее на woe-значениях считаем показатели:
- IV на трейне;
- Gini фич на трейне и тесте;
- Gini фич во времени (по неделям) на объединенной выборке (трейн+тест);
- PSI по неделям на объединенной выборке (трейн+тест), в качестве якорной выборки взять первую неделю в объединенном сэмпле;

7) **Предварительный отбор фичей**

Ограничения при отборе фич в модель:
- IV> 0.01;
- Gini > 0.03;
- Gini во времени стабилен и не падает до 0 ни в одном из периодов;
- PSI <= 0.10.

В случае несоблюдения ограничений – обосновать и задокументировать.

8) **Многофакторный анализ и отбор фичей в модель (не более 10 штук)**

- Найти методы отбора фич, дающие лучший результат (например см библиотеку *Mlxten, можете взять что-то другое);*
- Выбирать наилучшую комбинацию фичей удовлетворяющим условиям (Corr < 0.7 VIF < 10);
- Построить графики Gini модели на трейн и тесте в зависимости от кол-ва фич и их состава.

9) **На итоговом списке отобранных фичей построить итоговую моделей с оптимизацией гипер-параметров (gridsearch, optuna и тд).**
- контролить при этом качество (Gini) модели на размеченном тесте;
- все этапы документируем и сохраняем;
- модель пакуем в виде pickle/joblib.

10) **Оценить итоговой моделью все 3 выборки (трейн/тест и неразмеченную выборку) и посчитать тесты по модели:**

- на выборке (трейн+тест) посчитать индекс стабильности PSI по скору модели, разбитому на 10 бакетов ;
- биномиальный тест и тест Хосмер Лемешов на этих же 10 бакетах;
- Gini и Gini во времени аналогично как делали в переменных;
- передаем заказчику сэмплы и joblib/pickle файлы с преобразованиями, биннингом и моделями.

11) **Оформление подробного word-отчета с указанием всех шагов, графиками, таблицами со статистиками и документированием всех использованных скриптов.**

12) **Презентация с описанием проделанных шагов и тестов.**

13) **Передача документации и презентации заказчику.** Приложить также файл requirements.txt с использованными библиотеками и версиями python.

14) **Защита проекта.**

4. Дополнительная задача Black-box модель. (Опционально, но без решения основной задачи не принимается к рассмотрению)

Основные этапы по *моделям-кандидатам с использованием любых продвинутых методов (леса, бустинги, нейросети)*

1) Разбивка датасета на трейн/тест (тест и трейн из в основной задачи);
2) Препроцессинг фич под выбранный метод моделирования;
3) Однофакторный анализ, исходя из выбранного препроцессинга (если это возможно);
4) Модели-чемпионы (оценивать по тестовой выборке), оптимизация гипер-параметров, выбор итоговой black-box модели для передачи заказчику;
5) Shap/Lime/feature importance анализ;
7) Оценить моделью-чемпионом выборку df_test_notarget_10k.csv и передать отскоренную выборку заказчику;
8) Экспорт выбранной модели и препроцессинга в pickle/joblib файл (с указанием версий библиотек) и передача заказчику;
9) Короткий отчет.